List and local error-correction

Venkatesan Guruswami

Carnegie Mellon University

8th North American Summer School of Information Theory (NASIT)

San Diego, CA

August 11, 2015

Codes

Error-correcting code $C \subseteq \Sigma^N$
with encoding map $E : \mathcal{M} \to \Sigma^N$ $\quad$ (Image$(E) = C$)

- $\mathcal{M}$ = message space; $\quad \Sigma$ = alphabet; $\quad N$ = block length.
- To communicate *message* m, send **codeword** $E(m) \in C$.

Codes

Error-correcting code $C \subseteq \Sigma^N$
with encoding map $E : \mathcal{M} \to \Sigma^N \qquad (\text{Image}(E) = C)$

- $\mathcal{M}$ = message space; $\quad \Sigma$ = alphabet; $\quad N$ = block length.
- To communicate *message* m, send **codeword** $E(m) \in C$.

Rate $R = \frac{\log |\mathcal{M}|}{N \log |\Sigma|}$. $(\in [0, 1])$

- Ratio of # information bits communicated to # transmitted bits
- Identify messages $\mathcal{M} \simeq \Sigma^{RN}$; $\quad |C| = |\Sigma|^{RN}$.
- Proportion of redundant bits $= 1 - R$

Part I:

List decoding

Error correction

We'll be interested in correcting **worst-case** (adversarial) errors.

- arbitrary corruption of up to τN symbols (τ = error fraction)
- Both error locations and error values worst-case
- We count *symbol errors*, not bit errors.

Refer to τ as "decoding radius" (or error-correction radius)

Error correction

We'll be interested in correcting **worst-case** (adversarial) errors.

- arbitrary corruption of up to τN symbols (τ = error fraction)
- Both error locations and error values worst-case
- We count *symbol errors*, not bit errors.

Refer to τ as "decoding radius" (or error-correction radius)

Decoding problem for code $C \subset \Sigma^N$ up to radius τ:

Input: "Noisy received word" $\mathbf{y} \in \Sigma^N$

Output: Codeword $\mathbf{c} \in C$ such that the Hamming distance $\Delta(\mathbf{c}, \mathbf{y}) \leqslant \tau N$.

Rate vs. decoding radius

Goal

Would like *both* R and τ to be large (and alphabet Σ to be small).

(Think of $R, \tau \in (0,1)$ as fixed, and block length $N \to \infty$.)

Conflicting goals: correcting more errors requires more redundancy (lower rate).

Rate vs. error-correction radius

A trivial information-theoretic limit: $\tau \leqslant 1 - R$

- $|\mathcal{M}| = |\Sigma|^{RN} \implies$ need at least RN correct symbols from Σ to have any hope of meaningfully recovering message.
- Need *redundancy* $\geqslant$ *target error fraction.*

Question

Could we hope to approach such a nice trade-off?

Unique decoding

- $|C| = |\Sigma|^{RN} \Longrightarrow$ some two codewords $c_1 \neq c_2 \in C$ agree in first $RN - 1$ positions, i.e., differ in $\leqslant (1 - R)N + 1$ positions. *(Singleton Bound)*
- So when $\tau \geqslant (1 - R)/2$, can't unambigiously recover correct codeword (for worst-case errors).

Unique decoding

- $|C| = |\Sigma|^{RN} \implies$ some two codewords $c_1 \neq c_2 \in C$ agree in first $RN - 1$ positions, i.e., differ in $\leqslant (1 - R)N + 1$ positions. *(Singleton Bound)*
- So when $\tau \geqslant (1 - R)/2$, can't unambigiously recover correct codeword (for worst-case errors).

- "Unique decoding" for error fraction $\tau \approx (1 - R)/2$ achieved by Reed-Solomon (or similar) codes.
  - Note: This is over large alphabets; for binary codes, best possible trade-off unknown; Concatenated codes enable decoding up to Zyablov radius

- For larger τ, resort to **list decoding**.

List decoding

List decoding code $C \subset \Sigma^N$ up to radius τ:

Input: Noisy received word $\mathbf{y} \in \Sigma^N$

Output: A list of **all** codewords $\mathbf{c} \in C$ such that the Hamming distance $\Delta(\mathbf{c}, \mathbf{y}) \leqslant \tau N$.

List decoding

List decoding code $C \subset \Sigma^N$ up to radius τ:

Input: Noisy received word $\mathbf{y} \in \Sigma^N$

Output: A list of **all** codewords $\mathbf{c} \in C$ such that the Hamming distance $\Delta(\mathbf{c}, \mathbf{y}) \leqslant \tau N$.

Comments:

1. Code must guarantee that list is *small* for *every* $\mathbf{y}$
2. Need to find the list in $\mathrm{poly}(N)$ time, exploiting code structure.

A combinatorial definition

Definition (List decodability)

A code $C \subset \Sigma^N$ is said to be (τ, ℓ)-**list decodable** if for $\forall \mathbf{y} \in \Sigma^N$, there are $\leqslant \ell$ codewords of C within Hamming distance τN of $\mathbf{y}$.

Such a code offers potential for correcting τ fraction worst-case errors up to ambiguity ("list-size") ℓ.

The model of list decoding

But how useful is a list anyway?

1. List size > 1 typically a rare event (and we don't need to model channel stochastics precisely!)
2. In worst-case, better than decoding failure
   - Could use context/side information (or pick closest codeword) to disambiguate
3. Extensions such as list recovery & soft decoding very useful
   - decoding concatenated codes
   - practical use of channel reliability information
4. Versatile primitive
   - codes for computationally limited channels
5. Many applications beyond coding theory
   - eg. in complexity theory and cryptography
   - list decoding fits the bill as the right notion

The potential of list decoding

A code $C \subset \Sigma^N$ is said to be (τ, ℓ)-**list decodable** if for $\forall \mathbf{y} \in \Sigma^N$, there are $\leqslant \ell$ codewords of C within Hamming distance τN of y.

The potential of list decoding

A code $C \subset \Sigma^N$ is said to be (τ, ℓ)-**list decodable** if for $\forall \mathbf{y} \in \Sigma^N$, there are $\leqslant \ell$ codewords of C within Hamming distance τN of y.

Theorem (Non-constructive, via random coding)

For all $q \geqslant 2$, $\varepsilon > 0$ and $p \in (0, 1 - 1/q)$, there **exists** *a $(p, 1/\varepsilon)$-list decodable code of rate $1 - h_q(p) - \varepsilon$ over alphabet size q.*
($h_q(x) = x \log_q(q-1) - x \log_q x - (1-x) \log_q(1-x)$)

Conversely, **no** code of rate $1 - h_q(p) + \varepsilon$ is $(p, \exp(o(n)))$-list decodable.

The potential of list decoding

A code $C \subset \Sigma^N$ is said to be (τ, ℓ)-**list decodable** if for $\forall \mathbf{y} \in \Sigma^N$, there are $\leqslant \ell$ codewords of C within Hamming distance τN of y.

Theorem (Non-constructive, via random coding)

For all $q \geqslant 2$, $\varepsilon > 0$ and $p \in (0, 1 - 1/q)$, there **exists** *a $(p, 1/\varepsilon)$-list decodable code of rate $1 - h_q(p) - \varepsilon$ over alphabet size q.*
($h_q(x) = x \log_q(q-1) - x \log_q x - (1-x) \log_q(1-x)$)

Conversely, **no** code of rate $1 - h_q(p) + \varepsilon$ is $(p, \exp(o(n)))$-list decodable.

- Binary codes: Approach "Shannon capacity" of BSC_p for worst-case errors ("bridge" between Shannon & Hamming)

The potential of list decoding

A code $C \subset \Sigma^N$ is said to be (τ, ℓ)-**list decodable** if for $\forall \mathbf{y} \in \Sigma^N$, there are $\leqslant \ell$ codewords of C within Hamming distance τN of y.

Theorem (Non-constructive, via random coding)

For all $q \geqslant 2$, $\varepsilon > 0$ and $p \in (0, 1 - 1/q)$, there **exists** *a $(p, 1/\varepsilon)$-list decodable code of rate $1 - h_q(p) - \varepsilon$ over alphabet size q.*
($h_q(x) = x \log_q(q-1) - x \log_q x - (1-x) \log_q(1-x)$)

Conversely, **no** code of rate $1 - h_q(p) + \varepsilon$ is $(p, \exp(o(n)))$-list decodable.

- Binary codes: Approach "Shannon capacity" of BSC_p for worst-case errors ("bridge" between Shannon & Hamming)
- Large q: $(1 - R - \varepsilon, 1/\varepsilon)$-list decodable code over alphabet size $\exp(O(1/\varepsilon))$.

The potential of list decoding

A code $C \subset \Sigma^N$ is said to be (τ, ℓ)-**list decodable** if for $\forall \mathbf{y} \in \Sigma^N$, there are $\leqslant \ell$ codewords of C within Hamming distance τN of y.

Theorem (Non-constructive, via random coding)

For all $q \geqslant 2$, $\varepsilon > 0$ and $p \in (0, 1 - 1/q)$, there **exists** *a $(p, 1/\varepsilon)$-list decodable code of rate $1 - h_q(p) - \varepsilon$ over alphabet size q.*
($h_q(x) = x \log_q(q-1) - x \log_q x - (1-x) \log_q(1-x)$)

Conversely, **no** code of rate $1 - h_q(p) + \varepsilon$ is $(p, \exp(o(n)))$-list decodable.

- Binary codes: Approach "Shannon capacity" of BSC_p for worst-case errors ("bridge" between Shannon & Hamming)
- Large q: $(1 - R - \varepsilon, 1/\varepsilon)$-list decodable code over alphabet size $\exp(O(1/\varepsilon))$. $\Rightarrow$ **List decoding offers the potential to approach the $\tau = 1 - R$ limit with small list-size ℓ**

Random coding argument

Theorem

For all $q \geqslant 2$, $\varepsilon > 0$ and $p \in (0, 1 - 1/q)$, there **exists** *a $(p, 1/\varepsilon)$-list decodable code of rate $1 - h_q(p) - \varepsilon$ over alphabet size q.*

Proof sketch.

Let $R = 1 - h_q(p) - \varepsilon$ and $\ell = \frac{1}{\varepsilon} + 1$.
Pick q^{Rn} codewords at random from $\{1, 2, \ldots, q\}^n$.
Prob. that code is not $(p, \ell - 1)$-list decodable is at most

$$q^n \cdot q^{Rn\ell} \cdot \left(\frac{q^{h_q(p)n}}{q^n} \right)^{\ell}$$

Random coding argument

Theorem

For all $q \geqslant 2$, $\varepsilon > 0$ and $p \in (0, 1 - 1/q)$, there **exists** *a $(p, 1/\varepsilon)$-list decodable code of rate $1 - h_q(p) - \varepsilon$ over alphabet size q.*

Proof sketch.

Let $R = 1 - h_q(p) - \varepsilon$ and $\ell = \frac{1}{\varepsilon} + 1$.
Pick q^{Rn} codewords at random from $\{1, 2, \ldots, q\}^n$.
Prob. that code is not $(p, \ell - 1)$-list decodable is at most

$$q^n \cdot q^{Rn\ell} \cdot \left(\frac{q^{h_q(p)n}}{q^n} \right)^{\ell} = q^{n(1+\ell(R+h_q(p)-1))} = q^{n(1-\varepsilon\ell)} = q^{-\varepsilon n} \quad \square$$

Explicit list decoding

Challenges: Realize this constructively

1. List decode error fraction τ with an *explicit binary* code of rate $\approx 1 - h(\tau)$
2. List decode error fraction $\tau = 1 - R - \varepsilon$ with an *explicit* code of rate R

Explicit list decoding

Challenges: Realize this constructively

1. List decode error fraction τ with an *explicit binary* code of rate $\approx 1 - h(\tau)$
2. List decode error fraction $\tau = 1 - R - \varepsilon$ with an *explicit* code of rate R

The goal for binary codes is wide open.
But the second challenge over large alphabets has been met:

Explicit list decoding

Challenges: Realize this constructively

1. List decode error fraction τ with an *explicit binary* code of rate $\approx 1 - h(\tau)$
2. List decode error fraction $\tau = 1 - R - \varepsilon$ with an *explicit* code of rate R

The goal for binary codes is wide open.
But the second challenge over large alphabets has been met:

Theorem (G.-Rudra'08)

For all $R \in (0, 1)$ and $\varepsilon > 0$, explicit codes ("folded Reed-Solomon") of rate R with efficient list decoding up to radius $\tau = 1 - R - \varepsilon$.

Plus, subsequent improvements to other parameters (alphabet size, list-size, decoding complexity).

Rest of the talk

1. Decoding Reed-Solomon codes ✓
2. Folded Reed-Solomon codes: Linear-algebraic list decoding
3. Subspace-evasive pre-coding
4. Part II: Local Decoding

Reed-Solomon Codes

Definition (Reed-Solomon codes)

Messages = polynomials $f \in \mathbb{F}_q[X]$ of degree $< k$. Encoding:

$$f \mapsto (f(a_1), f(a_2), \ldots, f(a_n))$$

where $a_1, a_2, \ldots, a_n$ is a sequence of n distinct elements in $\mathbb{F}_q$ $(n \leqslant q)$.

Rate $= k/n$; Minimum distance $= n - k + 1$; alphabet size $= q$.

Classical algorithms to efficiently correct $\lfloor \frac{n-k}{2} \rfloor$ worst-case errors, starting with Peterson'60.
(note: error fraction $\tau = (1 - R)/2$)

RS unique decoding

Given n points $(a_i, y_i) \in \mathbb{F}_q^2$ s.t $\exists$ a (unique) polynomial $f \in \mathbb{F}_q[X]_{<k}$ with $f(a_i) \neq y_i$ for at most $e := \lfloor \frac{n-k}{2} \rfloor$ points
(so $f(a_i) = y_i$ for at least $\lceil \frac{n+k}{2} \rceil$ points).

Goal: find f.

RS unique decoding

Given n points $(a_i, y_i) \in \mathbb{F}_q^2$ s.t $\exists$ a (unique) polynomial $f \in \mathbb{F}_q[X]_{<k}$ with $f(a_i) \neq y_i$ for at most $e := \lfloor \frac{n-k}{2} \rfloor$ points
(so $f(a_i) = y_i$ for at least $\lceil \frac{n+k}{2} \rceil$ points).

Goal: find f.

Welch-Berlekamp algorithm

Let $D = \lceil \frac{n-k}{2} \rceil$

1. [Interpolation] Find a nonzero $Q(X, Y) = E(X)Y - N(X)$ with $\deg(E) \leqslant D$ and $\deg(N) < D + k$ s.t. $\forall i$, $Q(a_i, y_i) = 0$.
2. [Soluting finding] Output $N(X)/E(X)$

RS unique decoding

Given n points $(a_i, y_i) \in \mathbb{F}_q^2$ s.t $\exists$ a (unique) polynomial $f \in \mathbb{F}_q[X]_{<k}$ with $f(a_i) \neq y_i$ for at most $e := \lfloor \frac{n-k}{2} \rfloor$ points
(so $f(a_i) = y_i$ for at least $\lceil \frac{n+k}{2} \rceil$ points).

Goal: find f.

Welch-Berlekamp algorithm

Let $D = \lceil \frac{n-k}{2} \rceil$

1. [Interpolation] Find a nonzero $Q(X, Y) = E(X)Y - N(X)$ with $\deg(E) \leqslant D$ and $\deg(N) < D + k$ s.t. $\forall i$, $Q(a_i, y_i) = 0$.
2. [Soluting finding] Output $N(X)/E(X)$

Note: Such a $Q \neq 0$ always exists for *any* set of n points (a_i, y_i):

- $(D + 1) + D + k > 2D + k \geqslant n$ degrees of freedom, and n homogeneous linear constraints.

WB algorithm analysis

Assumption: $\exists f \in \mathbb{F}_q[X]_{<k}$ s.t. $f(a_i) = y_i$ for at least $\lceil \frac{n+k}{2} \rceil$ points)

Welch-Berlekamp algorithm

Let $D = \lceil \frac{n-k}{2} \rceil$

1. [Interpolation] Find a nonzero $Q(X, Y) = E(X)Y - N(X)$ with $\deg(E) \leqslant D$ and $\deg(N) < D + k$ s.t. $\forall i$, $Q(a_i, y_i) = 0$.
2. [Soluting finding] Output $N(X)/E(X)$

WB algorithm analysis

Assumption: $\exists f \in \mathbb{F}_q[X]_{<k}$ s.t. $f(a_i) = y_i$ for at least $\lceil \frac{n+k}{2} \rceil$ points)

Welch-Berlekamp algorithm

Let $D = \lceil \frac{n-k}{2} \rceil$

1. [Interpolation] Find a nonzero $Q(X, Y) = E(X)Y - N(X)$ with $\deg(E) \leqslant D$ and $\deg(N) < D + k$ s.t. $\forall i$, $Q(a_i, y_i) = 0$.
2. [Soluting finding] Output $N(X)/E(X)$

Claim: For *any* such Q, $f(X) = N(X)/E(X)$.

Proof: Define $R(X) = E(X)f(X) - N(X)$. By design, $\deg(R) < D + k = \lceil \frac{n+k}{2} \rceil$.
When $f(a_i) = y_i$, $R(a_i) = Q(a_i, f(a_i)) = Q(a_i, y_i) = 0$, so R has $\geqslant \lceil \frac{n+k}{2} \rceil$ roots.
Hence $R = 0$.

List decoding RS codes

Sudan's algorithm:

- Interpolating $Q(X, Y)$ with degree in Y equal to ℓ enables list decoding with list size ℓ (in second step find the at most ℓ "roots" $f(X)$ s.t. $Q(X, f(X)) = 0$)
- Working out parameters, can decode from agreement $> \frac{n}{\ell+1} + \frac{\ell(k-1)}{2}$
- Optimizing in ℓ, can correct $\approx n - \sqrt{2kn} = n(1 - \sqrt{2R})$ errors.

List decoding RS codes

Sudan's algorithm:

- Interpolating $Q(X, Y)$ with degree in Y equal to ℓ enables list decoding with list size ℓ (in second step find the at most ℓ "roots" $f(X)$ s.t. $Q(X, f(X)) = 0$)
- Working out parameters, can decode from agreement $> \frac{n}{\ell+1} + \frac{\ell(k-1)}{2}$
- Optimizing in ℓ, can correct $\approx n - \sqrt{2kn} = n(1 - \sqrt{2R})$ errors.

G.-Sudan algorithm:

- Use multiplicities in the interpolation.
- List decodes up to $n - \sqrt{kn} = (1 - \sqrt{R})n$ errors.
- Best known: RS codes not known to be *combinatorially* list-decodable with polysized lists beyond this radius.

Talk plan

1. Decoding Reed-Solomon codes
2. **Folded Reed-Solomon codes: Linear-algebraic list decoding**
3. Subspace-evasive pre-coding
4. Part II: Local decoding

Folded Reed-Solomon codes

Definition (Reed-Solomon codes)

Messages $=$ polynomials $f \in \mathbb{F}_q[X]$ of degree $< k$. Encoding:

$$f \mapsto (f(1), f(\gamma), f(\gamma^2), \ldots, f(\gamma^{n-1}))$$

where γ is a primitive element of $\mathbb{F}_q$ (and $n < q$).

Rate $= k/n$; alphabet size $= q$.

Folded Reed-Solomon codes

Definition (Reed-Solomon codes)

Messages = polynomials $f \in \mathbb{F}_q[X]$ of degree $< k$. Encoding:

$$f \mapsto (f(1), f(\gamma), f(\gamma^2), \ldots, f(\gamma^{n-1}))$$

where γ is a primitive element of $\mathbb{F}_q$ (and $n < q$).

Rate $= k/n$; alphabet size $= q$.

Definition (m-Folded Reed-Solomon codes)

Same rate; alphabet size q^m; block length $= n/m$

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

Folded Reed-Solomon list decoding

Theorem (G.-Rudra; via algebra of extension fields, building on **Parvaresh-Vardy**)

For any s, $1 \leqslant s \leqslant m$, the m-folded RS code can be list decoding from error fraction $\tau \approx 1 - \left(\frac{mR}{m-s+1}\right)^{s/(s+1)}$ with list-size q^s.

- $s = m = 1$ is the $1 - \sqrt{R}$ bound for RS codes.
- Picking $s \approx 1/\varepsilon$, $m \approx 1/\varepsilon^2$, $\tau \geqslant 1 - R - \varepsilon$.

Folded Reed-Solomon list decoding

Theorem (G.-Rudra; via algebra of extension fields, building on **Parvaresh-Vardy**)

For any s, $1 \leqslant s \leqslant m$, the m-folded RS code can be list decoding from error fraction $\tau \approx 1 - \left(\frac{mR}{m-s+1}\right)^{s/(s+1)}$ with list-size q^s.

- $s = m = 1$ is the $1 - \sqrt{R}$ bound for RS codes.
- Picking $s \approx 1/\varepsilon$, $m \approx 1/\varepsilon^2$, $\tau \geqslant 1 - R - \varepsilon$.

Theorem (Linear-algebra approach (G.-Wang'13))

For any s, $1 \leqslant s \leqslant m$, the m-folded RS code can be list decoded from error fraction $\tau = \frac{s}{s+1}\left(1 - \frac{mR}{m-s+1}\right)$ with list-size q^{s-1}.

- $s = m = 1$: $(1 - R)/2$ unique decoding bound for RS codes.
- Picking $s \approx 1/\varepsilon$, $m \approx 1/\varepsilon^2$, again $\tau \geqslant 1 - R - \varepsilon$.

Folded Reed-Solomon list decoding

- Following Reed-Solomon list decoder, two steps: (i) interpolation, and (ii) solution/root finding.

Folded Reed-Solomon list decoding

- Following Reed-Solomon list decoder, two steps:
  (i) interpolation, and (ii) solution/root finding.
- For folded codes, *multivariate* interpolation is used.
  In linear-algebraic version, interpolate a polynomial of following form [Vadhan] (for some $s \in \{1, 2, \ldots, m\}$)

$$A_0(X) + A_1(X)Y_1 + A_2(X)Y_2 + \cdots + A_s(X)Y_s$$

Folded Reed-Solomon list decoding

- Following Reed-Solomon list decoder, two steps:
  (i) interpolation, and (ii) solution/root finding.
- For folded codes, *multivariate* interpolation is used.
  In linear-algebraic version, interpolate a polynomial of following form [Vadhan] (for some $s \in \{1, 2, \ldots, m\}$)

$$A_0(X) + A_1(X)Y_1 + A_2(X)Y_2 + \cdots + A_s(X)Y_s$$

- Algebraic crux is to find all degree k solutions $f \in \mathbb{F}_q[X]$ to

$$A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) + \cdots + A_s(X)f(\gamma^{s-1}X) = 0$$

Folded Reed-Solomon list decoding

- Following Reed-Solomon list decoder, two steps:
  (i) interpolation, and (ii) solution/root finding.
- For folded codes, *multivariate* interpolation is used.
  In linear-algebraic version, interpolate a polynomial of following form [Vadhan] (for some $s \in \{1, 2, \ldots, m\}$)

$$A_0(X) + A_1(X)Y_1 + A_2(X)Y_2 + \cdots + A_s(X)Y_s$$

- Algebraic crux is to find all degree k solutions $f \in \mathbb{F}_q[X]$ to

$$A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) + \cdots + A_s(X)f(\gamma^{s-1}X) = 0$$

- Next: details of these steps
  - $s = 1$ corresponds to *unique* decoding: $f(X) = -A_0(X)/A_1(X)$.

Interpolation

$$\left(\begin{bmatrix} f(1) \\ f(\gamma) \\ \vdots \\ f(\gamma^{m-1}) \end{bmatrix}, \begin{bmatrix} f(\gamma^m) \\ f(\gamma^{m+1}) \\ \vdots \\ f(\gamma^{2m-1}) \end{bmatrix}, \ldots \right) \rightsquigarrow \left(\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{m-1} \end{bmatrix}, \ldots, \begin{bmatrix} y_{n-m} \\ y_{n-m+1} \\ \vdots \\ y_{n-1} \end{bmatrix} \right)$$

Find $A_0, A_1, \ldots, A_s \in \mathbb{F}_q[X]$ such that
$Q(X, Y_1, \cdots, Y_s) = A_0(X) + A_1(X)Y_1 + \cdots + A_s(X)Y_s$ satisfies

$$Q(\gamma^i, y_i, y_{i+1}, \ldots, y_{i+s-1}) = 0 \quad \forall\ i, i \mod m \in \{0, 1, \ldots, m-s\}\ .$$

Interpolation

$$\left(\begin{bmatrix} f(1) \\ f(\gamma) \\ \vdots \\ f(\gamma^{m-1}) \end{bmatrix}, \begin{bmatrix} f(\gamma^m) \\ f(\gamma^{m+1}) \\ \vdots \\ f(\gamma^{2m-1}) \end{bmatrix}, \ldots \right) \rightsquigarrow \left(\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{m-1} \end{bmatrix}, \ldots, \begin{bmatrix} y_{n-m} \\ y_{n-m+1} \\ \vdots \\ y_{n-1} \end{bmatrix} \right)$$

Find $A_0, A_1, \ldots, A_s \in \mathbb{F}_q[X]$ such that
$Q(X, Y_1, \cdots, Y_s) = A_0(X) + A_1(X)Y_1 + \cdots + A_s(X)Y_s$ satisfies

$$Q(\gamma^i, y_i, y_{i+1}, \ldots, y_{i+s-1}) = 0 \quad \forall\ i, i \mod m \in \{0, 1, \ldots, m-s\}\ .$$

- Restrict $\deg(A_0) < D + k$, $\deg(A_j) \leqslant D$ for $1 \leqslant j \leqslant s$.

Interpolation

$$\left(\begin{bmatrix} f(1) \\ f(\gamma) \\ \vdots \\ f(\gamma^{m-1}) \end{bmatrix}, \begin{bmatrix} f(\gamma^m) \\ f(\gamma^{m+1}) \\ \vdots \\ f(\gamma^{2m-1}) \end{bmatrix}, \ldots \right) \rightsquigarrow \left(\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{m-1} \end{bmatrix}, \ldots, \begin{bmatrix} y_{n-m} \\ y_{n-m+1} \\ \vdots \\ y_{n-1} \end{bmatrix} \right)$$

Find $A_0, A_1, \ldots, A_s \in \mathbb{F}_q[X]$ such that
$Q(X, Y_1, \cdots, Y_s) = A_0(X) + A_1(X)Y_1 + \cdots + A_s(X)Y_s$ satisfies

$$Q(\gamma^i, y_i, y_{i+1}, \ldots, y_{i+s-1}) = 0 \quad \forall\ i, i \mod m \in \{0, 1, \ldots, m-s\}\ .$$

- Restrict $\deg(A_0) < D + k$, $\deg(A_j) \leqslant D$ for $1 \leqslant j \leqslant s$.
- $> (s+1)D + k$ degrees of freedom/unknowns
- $n' := N(m - s + 1)$ constraints ($N = n/m$ is block length of folded code)

Linear interpolation step

Received word $\left(\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{m-1} \end{bmatrix}, \ldots, \begin{bmatrix} y_{n-m} \\ y_{n-m+1} \\ \vdots \\ y_{n-1} \end{bmatrix} \right)$

When $D = (n' - k)/(s+1)$, can find $A_0, A_1, \ldots, A_s \in \mathbb{F}_q[X]$, not all zero, such that
$Q(X, Y_1, \ldots, Y_s) = A_0(X) + A_1(X)Y_1 + \cdots + A_s(X)Y_s$ satisfies

1. $Q(\gamma^i, y_i, y_{i+1}, \ldots, y_{i+s-1}) = 0$ for $i \mod m \leqslant m - s$.
2. For any degree $< k$ polynomial f, $Q(X, f(X), f(\gamma X), \ldots, f(\gamma^{s-1}X))$ has degree $< D + k = (n' + sk)/(s+1)$

Second fact follows from degree restrictions on A_i's.

Algebraic handle on message polynomials

Lemma

If $t \geqslant \frac{n'+sk}{(m-s+1)(s+1)}$ *values of* $j \in \{0, 1, \ldots, N-1\}$ *satisfy* $(f(\gamma^{jm}), f(\gamma^{jm+1}), \ldots, f(\gamma^{jm+m-1})) = (y_{jm}, \ldots, y_{jm+m-1})$, *then* $A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) + \cdots + A_s(X)f(\gamma^{s-1}X) = 0$.

Algebraic handle on message polynomials

Lemma

If $t \geqslant \frac{n'+sk}{(m-s+1)(s+1)}$ *values of* $j \in \{0, 1, \ldots, N-1\}$ *satisfy* $(f(\gamma^{jm}), f(\gamma^{jm+1}), \ldots, f(\gamma^{jm+m-1})) = (y_{jm}, \ldots, y_{jm+m-1})$, *then* $A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) + \cdots + A_s(X)f(\gamma^{s-1}X) = 0$.

$$\left(\begin{bmatrix} f(1) \\ f(\gamma) \\ \vdots \\ f(\gamma^{m-1}) \end{bmatrix}, \ldots \right) \rightsquigarrow \left(\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{m-1} \end{bmatrix}, \ldots, \begin{bmatrix} y_{n-m} \\ y_{n-m+1} \\ \vdots \\ y_{n-1} \end{bmatrix} \right)$$

Key Fact: If codeword and $\mathbf{y}$ agree on t columns, then $(f(\gamma^i), f(\gamma^{i+1}), \ldots, f(\gamma^{i+s-1})) = (y_i, y_{i+1}, \ldots, y_{i+s-1})$ for at least $(m-s+1)t$ values of i.

The decoding radius

$N = n/m$ is block length of m-folded code.
$t = (1-\tau)N$ is the number of correct columns.

Decoding condition is

$$(1-\tau)N \geqslant \frac{N(m-s+1)+sk}{(s+1)(m-s+1)} = \frac{N}{s+1} + \frac{s}{s+1}\frac{m}{m-s+1}RN$$

(since $k = R \cdot n = R \cdot Nm$)

The decoding radius

$N = n/m$ is block length of m-folded code.
$t = (1-\tau)N$ is the number of correct columns.

Decoding condition is

$$(1-\tau)N \geqslant \frac{N(m-s+1)+sk}{(s+1)(m-s+1)} = \frac{N}{s+1} + \frac{s}{s+1}\frac{m}{m-s+1}RN$$

(since $k = R \cdot n = R \cdot Nm$) Above condition is met for

$$\tau \leqslant \frac{s}{s+1}\left(1 - \frac{m}{m-s+1}R\right) \ .$$

The decoding radius

$N = n/m$ is block length of m-folded code.
$t = (1-\tau)N$ is the number of correct columns.

Decoding condition is

$$(1-\tau)N \geqslant \frac{N(m-s+1)+sk}{(s+1)(m-s+1)} = \frac{N}{s+1} + \frac{s}{s+1}\frac{m}{m-s+1}RN$$

(since $k = R \cdot n = R \cdot Nm$) Above condition is met for

$$\tau \leqslant \frac{s}{s+1}\left(1 - \frac{m}{m-s+1}R\right) .$$

- Error fraction approches $\frac{s}{s+1}(1-R)$ for large $m \gg s$.
- Can achieve $\tau = 1 - R - \varepsilon$ by taking $s \gtrsim 1/\varepsilon$ and $m \gtrsim 1/\varepsilon^2$.

Recovering list of messages

Following interpolation step, algebraic crux is to find all degree $< k$ solutions $f \in \mathbb{F}_q[X]$ to the equation

$$A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) + \cdots + A_s(X)f(\gamma^{s-1}X) = 0$$

Recovering list of messages

Following interpolation step, algebraic crux is to find all degree $< k$ solutions $f \in \mathbb{F}_q[X]$ to the equation

$$A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) + \cdots + A_s(X)f(\gamma^{s-1}X) = 0$$

[G.'11] Observe that the above is an $\mathbb{F}_q$-linear system (in the coefficients of f)

- So we can solve for f and pin down possibilities to an affine subspace!
- To control list size, need to bound dimension of solution space.

Solving for f

Illustrate with $s = 2$

$$A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) = 0 \qquad (\clubsuit)$$

Let $A_i(X) = a_{i0} + a_{i1}X + a_{i2}X^2 + \cdots$ (wlog, not all $a_{i0} = 0$), and let $f = f_0 + f_1X + \cdots, +f_{k-1}X^{k-1}$.

Solving for f

Illustrate with $s = 2$

$$A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) = 0 \qquad (\clubsuit)$$

Let $A_i(X) = a_{i0} + a_{i1}X + a_{i2}X^2 + \cdots$ (wlog, not all $a_{i0} = 0$), and let $f = f_0 + f_1X + \cdots, +f_{k-1}X^{k-1}$.

($\clubsuit$) is the lower-triangular linear system:

$$\begin{aligned} a_{00} + (a_{10} + a_{20}) \cdot f_0 &= 0 \\ a_{01} + (\cdots) \cdot f_0 + (a_{10} + a_{20}\gamma) \cdot f_1 &= 0 \\ a_{02} + (\cdots) \cdot f_0 + (\cdots) \cdot f_1 + (a_{10} + a_{20}\gamma^2) \cdot f_2 &= 0 \\ &\vdots \end{aligned}$$

Solving for f

Illustrate with $s = 2$

$$A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) = 0 \qquad (\clubsuit)$$

Let $A_i(X) = a_{i0} + a_{i1}X + a_{i2}X^2 + \cdots$ (wlog, not all $a_{i0} = 0$), and let $f = f_0 + f_1X + \cdots, +f_{k-1}X^{k-1}$.

($\clubsuit$) is the lower-triangular linear system:

$$\begin{aligned} a_{00} + (a_{10} + a_{20}) \cdot f_0 &= 0 \\ a_{01} + (\cdots) \cdot f_0 + (a_{10} + a_{20}\gamma) \cdot f_1 &= 0 \\ a_{02} + (\cdots) \cdot f_0 + (\cdots) \cdot f_1 + (a_{10} + a_{20}\gamma^2) \cdot f_2 &= 0 \\ &\vdots \end{aligned}$$

At most one i s.t. $a_{10} + a_{20}\gamma^i = 0 \Longrightarrow$ soln. space dimension $\leqslant 1$.

Solving for f

Illustrate with $s = 2$

$$A_0(X) + A_1(X)f(X) + A_2(X)f(\gamma X) = 0 \qquad (\clubsuit)$$

Let $A_i(X) = a_{i0} + a_{i1}X + a_{i2}X^2 + \cdots$ (wlog, not all $a_{i0} = 0$), and let $f = f_0 + f_1X + \cdots, +f_{k-1}X^{k-1}$.

($\clubsuit$) is the lower-triangular linear system:

$$\begin{aligned} a_{00} + (a_{10} + a_{20}) \cdot f_0 &= 0 \\ a_{01} + (\cdots) \cdot f_0 + (a_{10} + a_{20}\gamma) \cdot f_1 &= 0 \\ a_{02} + (\cdots) \cdot f_0 + (\cdots) \cdot f_1 + (a_{10} + a_{20}\gamma^2) \cdot f_2 &= 0 \\ &\vdots \end{aligned}$$

At most one i s.t. $a_{10} + a_{20}\gamma^i = 0 \implies$ soln. space dimension $\leqslant 1$.
For general s, solns. lie in dim. $\leqslant s - 1$ subspace ($\therefore$ list size $\leqslant q^{s-1}$)

Summary

Folded RS decoding

For folded RS code of rate R, can list decode up to radius $\approx \frac{s}{s+1}(1-R)$ pinning down candidate messages to an affine subspace of dimension $\leqslant s-1$.

List size bound is q^{s-1}, or $q^{\Omega(1/\varepsilon)}$ when $s \approx 1/\varepsilon$.

Decoding complexity also similar, dominated by sifting through the $s-1$-dimensional subspace for close-by codewords.

Also $q > N$ (inherent to Reed-Solomon)

Analogous results for "derivative/multiplicity codes" [G.-Wang] [Kopparty]

Derivative/multiplicity codes

Definition (Order-m Derivative codes)

$a_1, a_2, \ldots, a_n$ distinct elements of $\mathbb{F}_q$, $\mathrm{char}(\mathbb{F}_q) > k$. Message $f \in \mathbb{F}_q[X]_{<k}$ is mapped to codeword

$$\left(\begin{bmatrix} f(a_1) \\ f'(a_1) \\ \vdots \\ f^{(m-1)}(a_1) \end{bmatrix}, \begin{bmatrix} f(a_2) \\ f'(a_2) \\ \vdots \\ f^{(m-1)}(a_2) \end{bmatrix}, \ldots, \begin{bmatrix} f(a_n) \\ f'(a_n) \\ \vdots \\ f^{(m-1)}(a_n) \end{bmatrix} \right).$$

Alphabet size q^m; block length $= n$; rate $R = k/(nm)$

For large $m \approx 1/\varepsilon^2$, can be list decoded from $1 - R - \varepsilon$ error fraction.

Summary

Optimal rate list decoding

Explicit (folded Reed-Solomon, or derivative) codes of rate R list-decodable up to error fraction $1 - R - \varepsilon$.

- Alphabet size $> N^{1/\varepsilon^2}$, and list-size $N^{1/\varepsilon}$.

Summary

Optimal rate list decoding

Explicit (folded Reed-Solomon, or derivative) codes of rate R list-decodable up to error fraction $1 - R - \varepsilon$.

- Alphabet size $> N^{1/\varepsilon^2}$, and list-size $N^{1/\varepsilon}$.

Codes also soft decodable and *list-recoverable*:

List recovery guarantee

Given sets $S_i \subseteq \Sigma$ with $|S_i| \leqslant \ell$ for each position $i \in [N]$, for appropriate parameters, there are at most $q^{\ell/\varepsilon}$ codewords c of folded RS code such that $c_i \in S_i$ for at least $(R + \varepsilon)N$ positions.

Key: Rate vs. decoding radius **not** impacted by ℓ; alphabet size $q^{O(\ell/\varepsilon^2)}$.

Binary list decoding

Optimal rate list recovery is very useful in concatenation schemes.

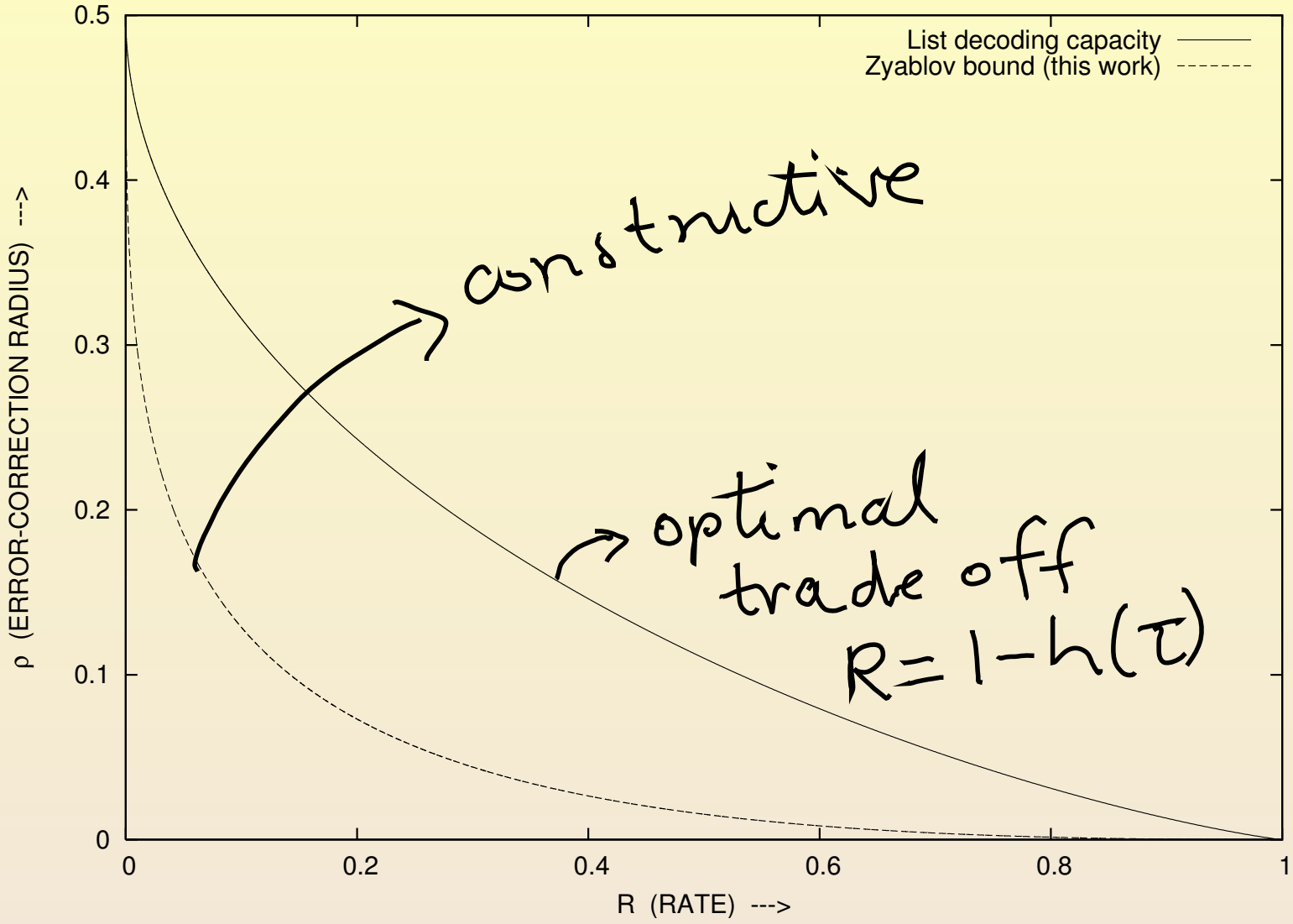
Folded Reed-Solomon codes concatenated with optimal short binary list-decodable codes $\Longrightarrow$ Binary codes for list decoding up to *Zyablov* bound:

$$\tau(R) = \max_{R_{\text{out}} R_{\text{in}} = R} \{(1 - R_{\text{out}}) h^{-1}(1 - R_{\text{in}})\}$$

Inner decoder passes sets S_i to outer folded RS list recovery algorithm; $\approx R_{\text{out}}$ of these sets containing the correct symbol.

Recall: List decoding "capacity" : $\tau^*(R) = h^{-1}(1 - R)$

Binary list decoding



Alphabet size reduction

Folded RS codes have alphabet size $> N^{1/\varepsilon^2}$.

Optimal alphabet size for list decoding up to $1 - R - \varepsilon$ error fraction is $\exp(O(1/\varepsilon))$ (also achieved by random codes)

[1](inspired by similar approach ideas used by [Alon-Luby'96] for erasure decoding, and [G.-Indyk'12] for error correction)

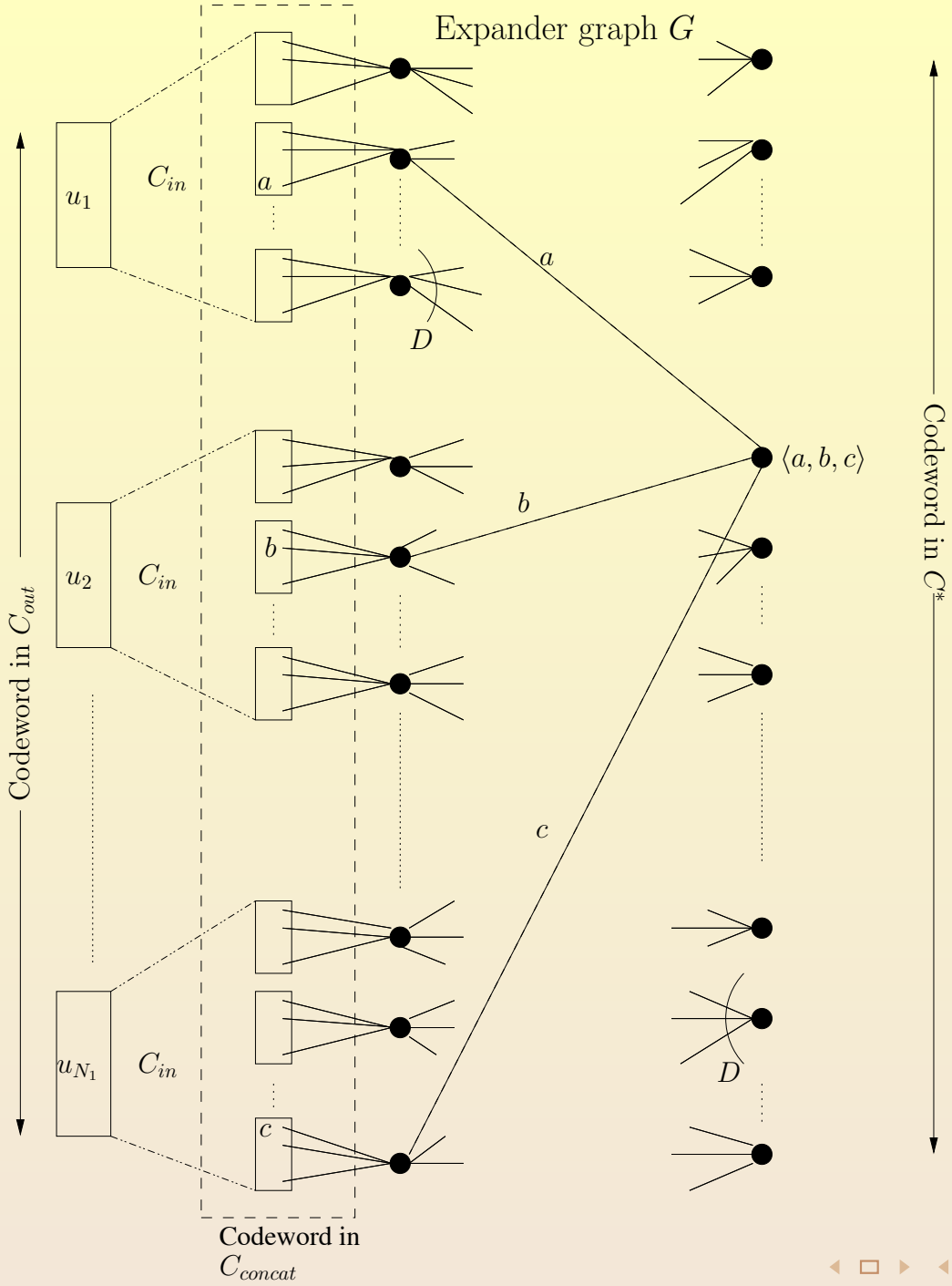
Alphabet size reduction

Folded RS codes have alphabet size $> N^{1/\varepsilon^2}$.

Optimal alphabet size for list decoding up to $1 - R - \varepsilon$ error fraction is $\exp(O(1/\varepsilon))$ (also achieved by random codes)

Using high rate folded RS codes in a concatenation scheme followed by expander graph based symbol redistribution[1]

[1] (inspired by similar approach ideas used by [Alon-Luby'96] for erasure decoding, and [G.-Indyk'12] for error correction)


Expander graph G
u_1
C_{in}
a
D
a
$\langle a, b, c \rangle$
b
u_2
C_{in}
b
c
u_{N_1}
C_{in}
c
D
Codeword in C_{out}
Codeword in C^*
Codeword in C_{concat}

Alphabet size reduction

Folded RS codes have alphabet size $> N^{1/\varepsilon^2}$.

Optimal alphabet size for list decoding up to $1 - R - \varepsilon$ error fraction is $\exp(O(1/\varepsilon))$ (also achieved by random codes)

Using high rate folded RS codes in a concatenation scheme followed by expander graph based symbol redistribution.

$\Rightarrow$ reduce alphabet size $\exp(1/\varepsilon^5)$ (**independent of block length**)

- But decoding complexity and list-size remain high ($> N^{1/\varepsilon}$), (inherited from outer folded RS code).

We turn to list-size reduction next...

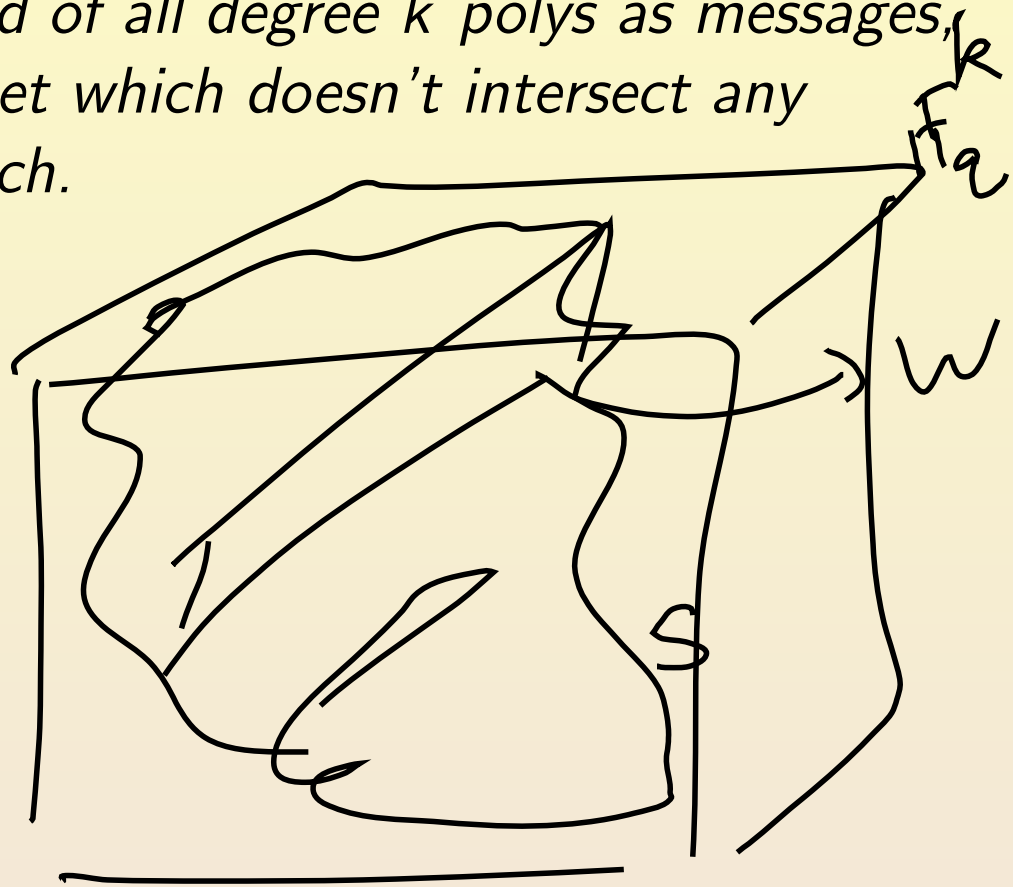
Talk plan

1. Decoding Reed-Solomon codes
2. Folded Reed-Solomon codes: Linear-algebraic list decoding
3. **Subspace-evasive pre-coding**
4. Part II: Local decoding

Pre-coding idea

In linear-algebraic list decoding, the list of candidate messages are contained within a s-dimensional subspace.

Simple yet influential idea: *Instead of all degree k polys as messages, only allow a carefully chosen subset which doesn't intersect any low-dimensional subspace too much.*



Pre-coding idea

In linear-algebraic list decoding, the list of candidate messages are contained within a s-dimensional subspace.

Simple yet influential idea: *Instead of all degree k polys as messages, only allow a carefully chosen subset which doesn't intersect any low-dimensional subspace too much.*

Subspace-evasive sets

A subset $S \subset \mathbb{F}_q^k$ is said to be (s, ℓ)-subspace evasive if for *all* s-dimensional subspaces W of $\mathbb{F}_q^k$, $|S \cap W| \leqslant \ell$.

Pre-coding idea

In linear-algebraic list decoding, the list of candidate messages are contained within a s-dimensional subspace.

Simple yet influential idea: *Instead of all degree k polys as messages, only allow a carefully chosen subset which doesn't intersect any low-dimensional subspace too much.*

Subspace-evasive sets

A subset $S \subset \mathbb{F}_q^k$ is said to be (s, ℓ)-subspace evasive if for *all* s-dimensional subspaces W of $\mathbb{F}_q^k$, $|S \cap W| \leqslant \ell$.

Observation: Restricting (coefficients of) message polynomials to belong to such a subspace-evasive set brings down list size to ℓ.

But how much does this cost in terms of rate?

Subspace-evasive sets

Natural notion (in pseudorandomness, geometry).

Considered in work on bipartite Ramsey problem [Pudlák-Rödl'05]

Subspace-evasive sets

Natural notion (in pseudorandomness, geometry).

Considered in work on bipartite Ramsey problem [Pudlák-Rödl'05]

Easy application of probabilistic method gives:

Lemma

A random subset of $\mathbb{F}_q^k$ of size $q^{(1-\varepsilon)k}$ is $(s, O(s/\varepsilon))$-subspace evasive w.h.p. (for $s \lesssim \varepsilon k$).

Factor $(1-\varepsilon)$ loss in rate suffices for significant pruning of the solution subspaces!

How to represent and encode into the subspace-evasive set?

Good subcodes of folded RS code

Prob. method works even for $O(s/\varepsilon)$-*wise independent subsets*, which admit compact representation and efficient encoding.

Via a pseudorandom construction of subspace-evasive sets, can get

- Monte Carlo consruction of a subcode of folded RS codes with list size $O(1/\varepsilon)$ (matching existential random coding bound!)

Upshot

Monte Carlo construction of efficiently $(1 - R - \varepsilon, O(1/\varepsilon))$-list decodable subcodes of folded Reed-Solomon codes.

Explicit construction?

Explicit subspace-evasive sets

Theorem (Dvir-Lovett'12)

Explicit *construction of a* $(s, (s/\varepsilon)^{O(s)})$*-subspace evasive subset of* $\mathbb{F}_q^k$ *of size* $q^{(1-\varepsilon)k}$.

Approach: An algberaic variety cut out by s polynomial equations such that the intersection with **every** s-dimensional affine space is a zero-dimensional variety. (Intersection size bound via *Bézout's theorem.*)

Explicit subspace-evasive sets

Theorem (Dvir-Lovett'12)

Explicit *construction of a* $(s, (s/\varepsilon)^{O(s)})$*-subspace evasive subset of* $\mathbb{F}_q^k$ *of size* $q^{(1-\varepsilon)k}$.

Approach: An algberaic variety cut out by s polynomial equations such that the intersection with **every** s-dimensional affine space is a zero-dimensional variety. (Intersection size bound via *Bézout's theorem.*)

Upshot

Explicit construction of efficiently $(1 - R - \varepsilon, \exp(\tilde{O}(1/\varepsilon)))$-list decodable codes.

Summary

Using subspace-evasive sets, we get subcodes of folded Reed-Solomon codes with rate R, list decoding radius $1 - R - \varepsilon$, list-size constant depending only on ε.

Summary

Using subspace-evasive sets, we get subcodes of folded Reed-Solomon codes with rate R, list decoding radius $1 - R - \varepsilon$, list-size constant depending only on ε.

Applying the (concatenation + expander-based) alphabet reduction ideas, we get

Optimal rate list decoding with fixed list & alphabet size

Explicit $(1 - R - \varepsilon, \exp(1/\varepsilon^{O(1)}))$-list decodable codes of rate R over alphabet size $\exp((1/\varepsilon^{O(1)}))$.

Summary

Using subspace-evasive sets, we get subcodes of folded Reed-Solomon codes with rate R, list decoding radius $1 - R - \varepsilon$, list-size constant depending only on ε.

Applying the (concatenation + expander-based) alphabet reduction ideas, we get

Optimal rate list decoding with fixed list & alphabet size

Explicit $(1 - R - \varepsilon, \exp(1/\varepsilon^{O(1)}))$-list decodable codes of rate R over alphabet size $\exp((1/\varepsilon^{O(1)}))$.

Some comments:

- Alphabet size not far from optimal; list size can be made $O(1/\varepsilon)$ with a Monte Carlo construction
- The alphabet reduction approach incurs large construction and decoding complexity of $N^{\mathrm{poly}(1/\varepsilon)}$

Folded Algebraic-geometric codes

Algebraic-geometric (AG) codes offer many of the desirable properties of RS codes, but over a smaller (even fixed) alphabet size.

Using appropriate automorphisms of the function field, one can "fold" AG codes; gives alternate approach to reduce alphabet size, without searching for optimal codes of length $(\log N)/\varepsilon^{O(1)}$.

Folded Algebraic-geometric codes

Algebraic-geometric (AG) codes offer many of the desirable properties of RS codes, but over a smaller (even fixed) alphabet size.

Using appropriate automorphisms of the function field, one can "fold" AG codes; gives alternate approach to reduce alphabet size, without searching for optimal codes of length $(\log N)/\varepsilon^{O(1)}$.

- [G.-Xing'12,'13] + [G.-Kopparty'13]: Explicit $(1 - R - \varepsilon, L)$-list decodable codes over alphabet of size $\exp(\tilde{O}(1/\varepsilon^2))$, with contruction/decoding complexity $O_\varepsilon(n^c)$
- List size L growing very slowly in the block length (like $\log^* N$)
- List size reduction achieved by subspace designs (a variant of subspace-evasive sets)

Wrap-up

- Variants of Reed-Solomon codes enable list decoding up to radius approaching optimal $1 - R$ bound with rate R.
- *Linear-algebraic approach* pins down candidates to a low-dimensional (or structured) subspace.

Wrap-up

- Variants of Reed-Solomon codes enable list decoding up to radius approaching optimal $1 - R$ bound with rate R.
- *Linear-algebraic approach* pins down candidates to a low-dimensional (or structured) subspace.
- Decoding approach versatile and applies to variants of
  - Algebraic-geometric codes (achieving constant alphabet size)
  - Gabidulin codes (optimal radius decoding in rank metric) and Koetter-Kschischang subspace codes [G.-Xing'13] [G.-Wang'14]
- Reduce list size by pruning subspace of candidate messages using (variants of) *subspace-evasive sets* or *subspace designs*.

Open Problems

- Large alphabet list decoding quite well understood.
  But $O_\varepsilon(n^c)$ complexity explicit $(1 - R - \varepsilon, L(\varepsilon))$-list decodable codes over alphabet size $Q(\varepsilon)$ open.
- List decoding capability of Reed-Solomon codes itself?
- Explicit optimal rate **binary** list-decodable codes?
  - Tackle case of erasures (list decoding up to $1 - R - \varepsilon$ erasure fraction with rate R)?
- Combinatorial bounds for list decoding
  - (τ, L)-list decodable binary code of rate $1 - h(\tau) - \varepsilon$:
    What's the smallest possible list-size $L = L(\varepsilon)$?
    We know $\log(1/\varepsilon) \lesssim L(\varepsilon) \lesssim 1/\varepsilon$

Part II:

Local decoding

Another strong decoding demand

Two relaxations of classical decoding against worst-case errors:

- List decoding demands correction of *more errors* beyond half-the-distance (even all the way up to the distance)
- Local decoding demands *highly efficient* correction of errors (a constant fraction of them, or even up to half-the-distance)

Another strong decoding demand

Two relaxations of classical decoding against worst-case errors:

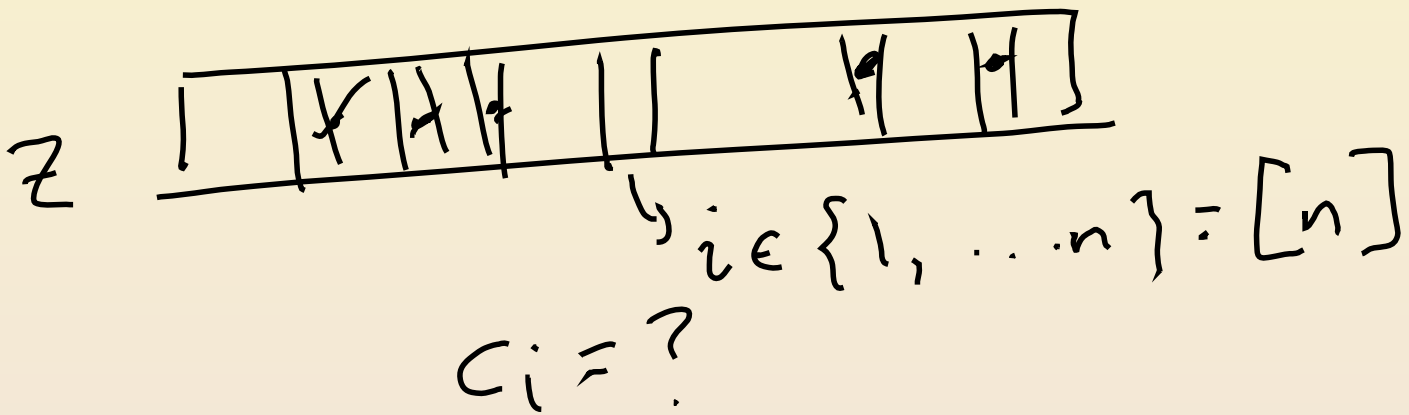
- List decoding demands correction of *more errors* beyond half-the-distance (even all the way up to the distance)
- Local decoding demands *highly efficient* correction of errors (a constant fraction of them, or even up to half-the-distance)

Note: One can also study *local list decoding*. Definition is a bit subtle and it has ramifications in average-case complexity and derandomization. But we won't study this model here.

Definition (Locally Correctable Codes (LCC))

For $\tau \in (0,1)$ and an integer r, a code $C \subseteq \Sigma^n$ is (r,τ)-**locally correctable** (or an (r,τ)-LCC) if there exists a randomized algorithm A, called a local corrector, satisfying the following requirements:

- **Input**: A takes as input coordinate $i \in [n]$ and gets oracle access to a string $z \in \Sigma^n$ with Hamming distance $\leqslant \tau n$ from some (unknown) codeword $c \in C$.
- **Output**: A outputs c_i with probability at least $3/4$.
- **Query efficiency**: A makes at most r queries to the oracle z.



Definition (Locally Correctable Codes (LCC))

For $\tau \in (0,1)$ and an integer r, a code $C \subseteq \Sigma^n$ is (r,τ)-**locally correctable** (or an (r,τ)-LCC) if there exists a randomized algorithm A, called a local corrector, satisfying the following requirements:

- **Input**: A takes as input coordinate $i \in [n]$ and gets oracle access to a string $z \in \Sigma^n$ with Hamming distance $\leqslant \tau n$ from some (unknown) codeword $c \in C$.
- **Output**: A outputs c_i with probability at least $3/4$.
- **Query efficiency**: A makes at most r queries to the oracle z.

Goal is to have $r \ll n$, say n^{ε} or perhaps even $O(1)$.

Note: One can also demand that A runs in time $\mathrm{poly}(r)$.
This is the case for known constructions, but for simplicity we will focus on trade-off between query complexity r and redundancy of code.

LCC: State of the art

For simplicity, let's focus on two extreme regimes

- Low query complexity (and high redundancy)
- Low redundancy (and yet sub-linear query complexity)

(and let's settle for any τ bounded away from 0 for now, and suppress its mention)

LCC: State of the art

$r=3$

$n = 2^{O(\sqrt{k})}$

For simplicity, let's focus on two extreme regimes

- Low query complexity (and high redundancy)
- Low redundancy (and yet sub-linear query complexity)

(and let's settle for any τ bounded away from 0 for now, and suppress its mention)

Constant query complexity

For $r = O(1)$, there are r-query LCCs with $n = \exp(k^{\frac{1}{r-1}})$ (k =dimension of the code).

In fact, these are just good old (non-binary) Reed-Muller codes.

Best lower bound is only $n \geqslant \Omega(k^2)$ even for $r = 3$.

LCC: State of the art

Constant query complexity

For $r = O(1)$, there are r-query LCCs (appropriate Reed-Muller codes) with $n = \exp(k^{\frac{1}{r-1}})$ (k =dimension of the code).

LCC: State of the art

Constant query complexity

For $r = O(1)$, there are r-query LCCs (appropriate Reed-Muller codes) with $n = \exp(k^{\frac{1}{r-1}})$ (k =dimension of the code).

High rate

For any $\varepsilon > 0$, there are rate $R = (1 - \varepsilon)$ LCCs with $r \leqslant n^{o(1)}$ (specifically $r \leqslant 2^{O(\sqrt{\log n \log\log n})}$)

LCC: State of the art

Constant query complexity

For $r = O(1)$, there are r-query LCCs (appropriate Reed-Muller codes) with $n = \exp(k^{\frac{1}{r-1}})$ (k =dimension of the code).

High rate

For any $\varepsilon > 0$, there are rate $R = (1 - \varepsilon)$ LCCs with $r \leqslant n^{o(1)}$ (specifically $r \leqslant 2^{O(\sqrt{\log n \log\log n})}$)

- Till 2010, for rate $> 1/2$, no $o(n)$-query LCC known!
- Can get $\tau \approx (1 - R)/2$ over large alphabets and $\tau \approx$ half-the-Zyablov-bound in binary case
- Lower bound is only $r \geqslant \Omega(\log n)$.

Locally decodable codes

Demand local recovery of only the "information symbols".

Definition (LDC)

An encoder $C : \Sigma^k \to \Sigma^n$ is an (r, τ)-LDC if there is a randomized algorithm A satisfying the following requirements:

- **Input**: A takes as input coordinate $i \in [k]$ and gets oracle access to a string $z \in \Sigma^n$ with Hamming distance $\leqslant \tau n$ from $C(x)$ for some (unknown) $x \in \Sigma^k$.
- **Output**: A outputs x_i with probability at least $3/4$.
- **Query efficiency**: A makes at most r queries to the oracle z.

Note

By choosing a systematic encoding map, a *linear* LCC yields an LDC with same rate, τ, and query complexity r.

Matching Vector LDC

Till 2007, best known LDCs were also Reed-Muller codes; in particular, $r = O(1)$ required encoding length $n \geqslant \exp(k^{\Omega(1)})$ (as it *still the case* for LCCs).

[Yekhanin'07]: surprising LDCs using "matching vector families"

Sub-exponential length LDCs

[Yekhanin-Raghavendra-Efremenko] 3-query LDCs with $n \leqslant \exp(k^{o(1)})$, specifically $n \leqslant \exp(2^{O(\sqrt{\log k \log\log k})})$.

$\hookrightarrow \exp(\sqrt{k})$

Matching Vector LDC

Till 2007, best known LDCs were also Reed-Muller codes; in particular, $r = O(1)$ required encoding length $n \geqslant \exp(k^{\Omega(1)})$ (as it *still the case* for LCCs).

[Yekhanin'07]: surprising LDCs using "matching vector families"

Sub-exponential length LDCs

[Yekhanin-Raghavendra-Efremenko] 3-query LDCs with $n \leqslant \exp(k^{o(1)})$, specifically $n \leqslant \exp(2^{O(\sqrt{\log k \log\log k})})$.

Can be described in terms of "log generator matrix" $\in \mathbb{Z}_6^{k \times n}$ whose columns are element of a $\mathbb{Z}_6$-module with a "nice" $k \times k$ minor.

Construction of such a matrix not too hard to describe (using low-degree polynomials representing OR mod 6), but its applicability to LDC is still mysterious to me.

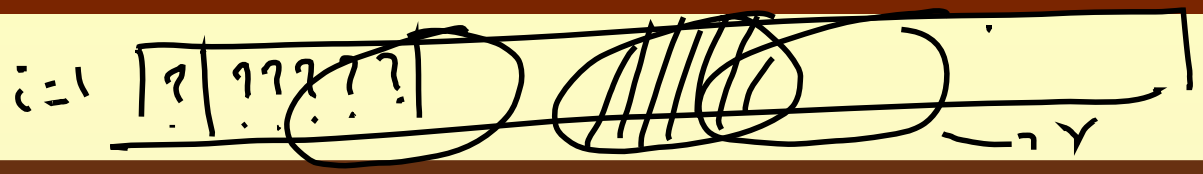
Erasure recovery

For simplicity, let's restrict attention to erasure model: arbitrary τ fraction of symbols are erased.

Constructions are the same as for errors, and analysis ingredients are similar, so good way to illustrate the main ideas of the theory.

For erasure case, can have a more crisp, combinatorial definition.

Locally Erasure Correctable Codes



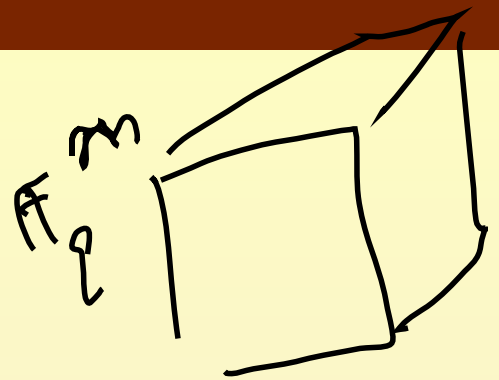
Definition (Erasure LCC)

For $\tau \in (0,1)$ and an integer r, a code $C \subseteq \Sigma^n$ is a (r,τ) (erasure) LCC if for every $i \in [n]$, there is a family $\mathcal{F}_i$ of r-element subsets of $[n] \setminus \{i\}$ such that

- For each $S \in \mathcal{F}$ and $c \in C$, c_i is determined by $\{c_j \mid j \in S\}$
- For all $i \in [n]$ and erasure locations $T \subseteq [n]$ of size $|T| = \tau n$, there is some $S \in \mathcal{F}_i$ s.t $S \cap T = \emptyset$.

For linear codes, we are simply asking for a weight $r+1$ dual codeword whose support contains i and that avoids the (arbitrarily chosen) τn erasure locations.

Reed-Muller codes



For a field $\mathbb{F}_q$ and degree $d < q$,

$$\mathrm{RM}_q(m,d) = \{\langle f(\mathbf{a})\rangle_{\mathbf{a}\in\mathbb{F}_q^m} \mid f \in \mathbb{F}_q[X_1, X_2, \ldots, X_m], \quad \deg(f) \leqslant d\}$$

(evaluations of total degree $\leqslant d$ m-variate polys on all points in $\mathbb{F}_q^m$)

Fact

For $d < q$, $\mathrm{RM}_q(m,d)$ is an $[q^m, \binom{m+d}{d}, \left(1 - \frac{d}{q}\right) q^m]$ $\mathbb{F}_q$-linear code.

Theorem (Reed-Muller LCC)

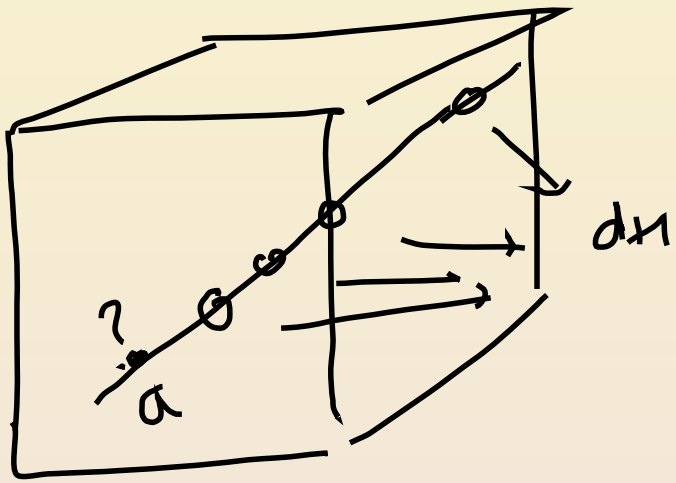
For $r = O(1)$, there are r-query LCCs with $n = \exp(O(k^{\frac{1}{r-1}}))$ (k =dimension of the code).

Theorem (Reed-Muller LCC)

For $r = O(1)$, there are r-query LCCs with $n = \exp(O(k^{\frac{1}{r-1}}))$ (k =dimension of the code).

Key Local Correction Property

On each line $\ell_{\mathbf{a},\mathbf{b}} = \{\mathbf{a} + \lambda\mathbf{b} \mid \lambda \in \mathbb{F}_q\}$ (here $\mathbf{a}, \mathbf{b} \in \mathbb{F}_q^m$),
f restricted to $\ell_{\mathbf{a},\mathbf{b}}$ is a degree d polynomial in λ.
$\implies$ $f(\mathbf{a})$ can be recovered from any $d+1$ values of f on $\ell_{\mathbf{a},\mathbf{b}}$.



Theorem (Reed-Muller LCC)

For $r = O(1)$, there are r-query LCCs with $n = \exp(O(k^{\frac{1}{r-1}}))$ (k =dimension of the code).

Key Local Correction Property

On each line $\ell_{\mathbf{a},\mathbf{b}} = \{\mathbf{a} + \lambda\mathbf{b} \mid \lambda \in \mathbb{F}_q\}$ (here $\mathbf{a}, \mathbf{b} \in \mathbb{F}_q^m$),
f restricted to $\ell_{\mathbf{a},\mathbf{b}}$ is a degree d polynomial in λ.
$\implies$ $f(\mathbf{a})$ can be recovered from any $d+1$ values of f on $\ell_{\mathbf{a},\mathbf{b}}$.

Assume erasure fraction $\tau < 1/4$ (for simplicity).
Parameter choices for $\mathrm{RM}_q(m, d)$ to get (r, τ) (erasure) LCC:

- $d = r - 1$ $\quad$ ($\because$ r queries in $\ell_{\mathbf{a},\mathbf{b}}$ suffice to determine $f(\mathbf{a})$)

Theorem (Reed-Muller LCC)

For $r = O(1)$, there are r-query LCCs with $n = \exp(O(k^{\frac{1}{r-1}}))$ (k =dimension of the code).

Key Local Correction Property

On each line $\ell_{\mathbf{a},\mathbf{b}} = \{\mathbf{a} + \lambda\mathbf{b} \mid \lambda \in \mathbb{F}_q\}$ (here $\mathbf{a}, \mathbf{b} \in \mathbb{F}_q^m$),
f restricted to $\ell_{\mathbf{a},\mathbf{b}}$ is a degree d polynomial in λ.
$\implies$ $f(\mathbf{a})$ can be recovered from any $d+1$ values of f on $\ell_{\mathbf{a},\mathbf{b}}$.

Assume erasure fraction $\tau < 1/4$ (for simplicity).
Parameter choices for $\mathrm{RM}_q(m, d)$ to get (r, τ) (erasure) LCC:

- $d = r - 1$ ($\because$ r queries in $\ell_{\mathbf{a},\mathbf{b}}$ suffice to determine $f(\mathbf{a})$)
- $q \geqslant 2d$ ($\because$ average line has $(1-\tau)q \geqslant \frac{3q}{4} \geqslant d+1$ unerased coordinates)

Theorem (Reed-Muller LCC)

There are r-query LCCs with $n = \exp(O_r(k^{\frac{1}{r-1}}))$ *(*k *=dimension of the code).*

Recall $d = r - 1$, $q \approx 2d$.

- Code length $n = q^m \leqslant (2r)^m$.
- Dimension $k = \binom{m+d}{d} \geqslant \left(\frac{m}{d}\right)^d \geqslant \left(\frac{m}{r}\right)^{r-1}$

Theorem (Reed-Muller LCC)

There are r-query LCCs with $n = \exp(O_r(k^{\frac{1}{r-1}}))$ (k =dimension of the code).

Recall $d = r - 1$, $q \approx 2d$.

$m \leq r \cdot k^{\frac{1}{r-1}}$

- Code length $n = q^m \leqslant (2r)^m$.
- Dimension $k = \binom{m+d}{d} \geqslant \left(\frac{m}{d}\right)^d \geqslant \left(\frac{m}{r}\right)^{r-1}$
- $\therefore$ ‘ $n \leqslant (2r)^{r \cdot k^{1/(r-1)}} \leqslant \exp(O_r(k^{\frac{1}{r-1}}))$.

Theorem (Reed-Muller LCC)

There are r-query LCCs with $n = \exp(O_r(k^{\frac{1}{r-1}}))$ *(*k *=dimension of the code).*

Recall $d = r - 1$, $q \approx 2d$.

- Code length $n = q^m \leqslant (2r)^m$.
- Dimension $k = \binom{m+d}{d} \geqslant \left(\frac{m}{d}\right)^d \geqslant \left(\frac{m}{r}\right)^{r-1}$
- $\therefore$ ‘ $n \leqslant (2r)^{r \cdot k^{1/(r-1)}} \leqslant \exp(O_r(k^{\frac{1}{r-1}}))$.

A different regime of choices: constant number of variables m, say $m = 1/\varepsilon$.

- number of queries $r \approx n^{1/m} = n^{\varepsilon}$
- dimension $k = \binom{m+d}{m} \geqslant \left(\frac{d}{m}\right)^m \geqslant \left(\frac{q}{2m}\right)^m = \frac{n}{(2m)^m} \geqslant \varepsilon^{O(1/\varepsilon)} \cdot n$

Rered-Muller LCC trade-offs

1. Locality $r = O(1)$, $n = \exp(O(k^{1/(r-1)}))$.
2. Locality $r = n^{\varepsilon}$, rate $= \varepsilon^{O(1/\varepsilon)}$

Rered-Muller LCC trade-offs

1. Locality $r = O(1)$, $n = \exp(O(k^{1/(r-1)}))$.
2. Locality $r = n^{\varepsilon}$, rate $= \varepsilon^{O(1/\varepsilon)}$

Initial beliefs, may be these are best possible:

- Locality $O(1)$ requires $n = \exp(k^{\Omega(1)})$?
- Locality n^{ε} for $\varepsilon \to 0$ require rate $\to 0$?

Latter disproved (in *3* different ways!): can even have rate $\to 1$ together with locality n^{ε}.

Former disproved for *LDCs*, jury still out in the case of LCCs.

High rate LCCs

[Kopparty, Meir, Ron-Zewi, Saraf'15]; multiplicity codes + expander

For any $\gamma > 0$, there are rate $(1-\gamma)$ LCCs with locality $r \leqslant n^{o(1)}$ (specifically $r \leqslant 2^{O(\sqrt{\log n \log\log n})}$)

High rate LCCs

[Kopparty, Meir, Ron-Zewi, Saraf'15]; multiplicity codes + expander

For any $\gamma > 0$, there are rate $(1-\gamma)$ LCCs with locality $r \leqslant n^{o(1)}$ (specifically $r \leqslant 2^{O(\sqrt{\log n \log\log n})}$)

Let's settle for high rate and locality $r \leqslant n^{\varepsilon}$ for constant $\varepsilon > 0$.

Achieved by *three* recent constructions (before which even locality $o(n)$ required rate $< 1/2$):

1. Multiplicity codes [Kopparty, Saraf, Yekhanin'11]
2. Lifted codes [Guo, Sudan]
3. Expander based codes [Hemenway, Ostrovsky, Wootters'14]

We'll only discuss multiplicity codes.

Reed-Muller rate limitation

Consider $\mathrm{RM}_q(m, d)$ for $m > 1$ and $d < q$.

Rate is maximized for $m = 2$, in which case it is

$$\frac{\binom{d+2}{2}}{q^2} \leqslant \frac{1}{2} .$$

So need different codes to go beyond rate barrier.

Reed-Muller rate limitation

Consider $\mathrm{RM}_q(m, d)$ for $m > 1$ and $d < q$.

Rate is maximized for $m = 2$, in which case it is

$$\frac{\binom{d+2}{2}}{q^2} \leqslant \frac{1}{2} .$$

So need different codes to go beyond rate barrier.

Lifted codes:

- Take codes that fit the algorithm! Only need restrictions on lines to be Reed-Solomon codes.
- Same local correction algorithm as RM codes. Non-trivial analysis to show high rate.

Multiplicity codes

To improve rate of RM codes, include not just evaluations of polynomials but also its partial derivatives (of order $< s$).

- A priori it seems this would hurt the rate (more redundant encoding)

Multiplicity codes

To improve rate of RM codes, include not just evaluations of polynomials but also its partial derivatives (of order $< s$).

- A priori it seems this would hurt the rate (more redundant encoding)
- But can increase the degree of message polynomials to compensate (and in fact gain in rate)
- Illustrate for $s = 2$ and bivariate case (general case similar)

Multiplicity codes

To improve rate of RM codes, include not just evaluations of polynomials but also its partial derivatives (of order $< s$).

- A priori it seems this would hurt the rate (more redundant encoding)
- But can increase the degree of message polynomials to compensate (and in fact gain in rate)
- Illustrate for $s = 2$ and bivariate case (general case similar)

Caveat

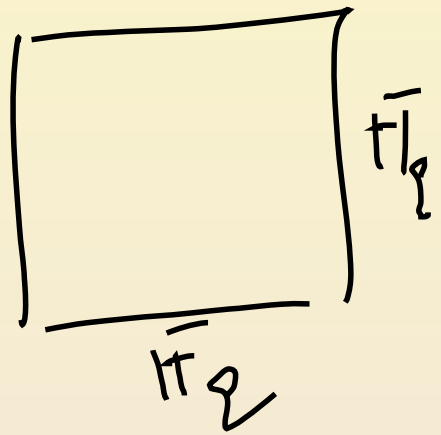
Derivatives over small characteristic are troublesome; one can use Hasse derivatives in such cases. We'll ignore this technical issue, and assume characteristic is large enough that formal partial derivatives behave like they should.

Multiplicity codes

Definition (Bivariate multiplicity 2 codes)

- Messages = polynomials $f \in \mathbb{F}_q[X_1, X_2]$ of total degree $\leqslant d$
- Encoding = $\langle f(\mathbf{a}), \frac{\partial f}{\partial X_1}(\mathbf{a}), \frac{\partial f}{\partial X_2}(\mathbf{a}) \rangle_{\mathbf{a} \in \mathbb{F}_q \times \mathbb{F}_q}$
- Block length = q^2; Code alphabet = $\mathbb{F}_q^3$

- Rate of code



Multiplicity codes

Definition (Bivariate multiplicity 2 codes)

- Messages $=$ polynomials $f \in \mathbb{F}_q[X_1, X_2]$ of total degree $\leqslant d$
- Encoding $= \langle f(\mathbf{a}), \frac{\partial f}{\partial X_1}(\mathbf{a}), \frac{\partial f}{\partial X_2}(\mathbf{a}) \rangle_{\mathbf{a} \in \mathbb{F}_q \times \mathbb{F}_q}$
- Block length $= q^2$; Code alphabet $= \mathbb{F}_q^3$

- Rate of code $= \frac{\binom{d+2}{2}}{3q^2}$
- Factor 3 smaller than corresponding Reed-Muller code ??

Multiplicity codes

Definition (Bivariate multiplicity 2 codes)

- Messages $=$ polynomials $f \in \mathbb{F}_q[X_1, X_2]$ of total degree $\leqslant d$
- Encoding $= \langle f(\mathbf{a}), \frac{\partial f}{\partial X_1}(\mathbf{a}), \frac{\partial f}{\partial X_2}(\mathbf{a}) \rangle_{\mathbf{a} \in \mathbb{F}_q \times \mathbb{F}_q}$
- Block length $= q^2$; Code alphabet $= \mathbb{F}_q^3$

- Rate of code $= \frac{\binom{d+2}{2}}{3q^2}$
- Factor 3 smaller than corresponding Reed-Muller code ??
- Relative distance $\geqslant 1 - \frac{d}{2q}$ ($\because$ one picks up multiplicity **2** root when partials also vanish)

Multiplicity codes

Definition (Bivariate multiplicity 2 codes)

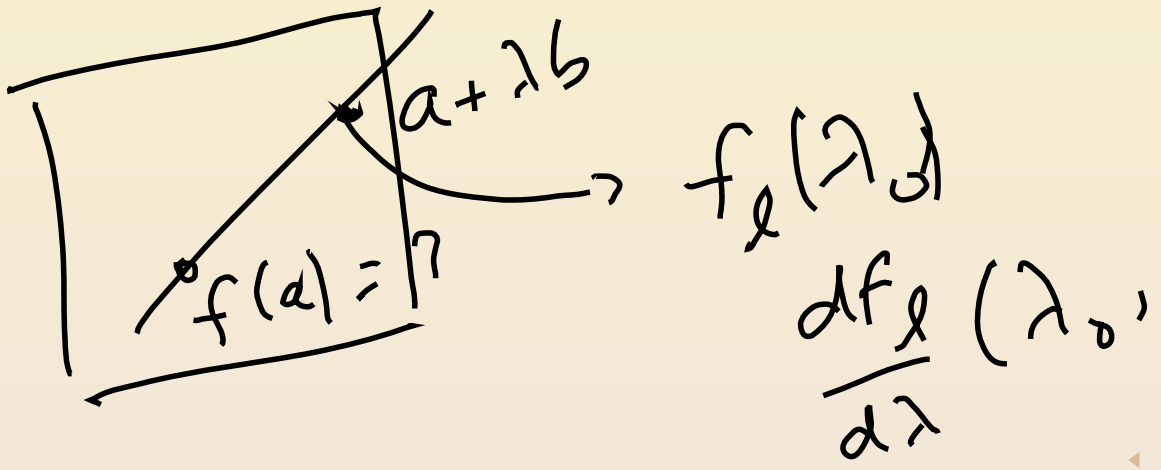
- Messages = polynomials $f \in \mathbb{F}_q[X_1, X_2]$ of total degree $\leqslant d$
- Encoding = $\langle f(\mathbf{a}), \frac{\partial f}{\partial X_1}(\mathbf{a}), \frac{\partial f}{\partial X_2}(\mathbf{a}) \rangle_{\mathbf{a} \in \mathbb{F}_q \times \mathbb{F}_q}$
- Block length = q^2; Code alphabet = $\mathbb{F}_q^3$

- Rate of code = $\frac{\binom{d+2}{2}}{3q^2}$ $\simeq$ $\frac{d^2}{6q^2} \approx \frac{4}{6} = \frac{2}{3}$
- Factor 3 smaller than corresponding Reed-Muller code ??
- Relative distance $\geqslant 1 - \frac{d}{2q}$ ($\because$ one picks up multiplicity **2** root when partials also vanish)
- So can take twice the degree!
- Taking $d \approx 2q$, rate approaches $2/3$

Local correction algorithm

Given order-2 multplicity encoding of degree d poly $f \in \mathbb{F}_q[X_1, X_2]$, with τ fraction of erasures; assume $\tau < \frac{1}{2}\left(1 - \frac{d}{2q}\right)$.

Local recovery of erased symbol at position $\mathbf{a} \in \mathbb{F}_q^2$



Local correction algorithm

Given order-2 multplicity encoding of degree d poly $f \in \mathbb{F}_q[X_1, X_2]$, with τ fraction of erasures; assume $\tau < \frac{1}{2}\left(1 - \frac{d}{2q}\right)$.

Local recovery of erased symbol at position $\mathbf{a} \in \mathbb{F}_q^2$

1. Pick a line $\ell = \{\mathbf{a} + \lambda \mathbf{b} \mid \lambda \in \mathbb{F}_q\}$ through a that has $\leqslant 2\tau q$ erasures ($\geqslant 1/2$ the lines are such)

Local correction algorithm

Given order-2 multplicity encoding of degree d poly $f \in \mathbb{F}_q[X_1, X_2]$, with τ fraction of erasures; assume $\tau < \frac{1}{2}\left(1 - \frac{d}{2q}\right)$.

Local recovery of erased symbol at position $\mathbf{a} \in \mathbb{F}_q^2$

1. Pick a line $\ell = \{\mathbf{a} + \lambda \mathbf{b} \mid \lambda \in \mathbb{F}_q\}$ through a that has $\leqslant 2\tau q$ erasures ($\geqslant 1/2$ the lines are such)
2. Interpolate $f_\ell \in \mathbb{F}_q[\lambda]$ of degree d from values of f_ℓ and its derivative $\frac{df_\ell}{d\lambda}$ at the ($> d/2$) unerased points in ℓ.

Chain rule

$$\frac{df_\ell}{d\lambda}(\lambda_0) = \frac{\partial f}{\partial X_1}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_1 + \frac{\partial f}{\partial X_2}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_2$$

Local correction algorithm

Given order-2 multplicity encoding of degree d poly $f \in \mathbb{F}_q[X_1, X_2]$, with τ fraction of erasures; assume $\tau < \frac{1}{2}\left(1 - \frac{d}{2q}\right)$.

Local recovery of erased symbol at position $\mathbf{a} \in \mathbb{F}_q^2$

1. Pick a line $\ell = \{\mathbf{a} + \lambda \mathbf{b} \mid \lambda \in \mathbb{F}_q\}$ through a that has $\leqslant 2\tau q$ erasures ($\geqslant 1/2$ the lines are such)
2. Interpolate $f_\ell \in \mathbb{F}_q[\lambda]$ of degree d from values of f_ℓ and its derivative $\frac{df_\ell}{d\lambda}$ at the ($> d/2$) unerased points in ℓ.

Chain rule

$$\frac{df_\ell}{d\lambda}(\lambda_0) = \frac{\partial f}{\partial X_1}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_1 + \frac{\partial f}{\partial X_2}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_2$$

Now $f_\ell(0) = f(\mathbf{a})$, so value of f at $\mathbf{a}$ can be recovered.

Local correction algorithm

Given order-2 multplicity encoding of degree d poly $f \in \mathbb{F}_q[X_1, X_2]$, with τ fraction of erasures; assume $\tau < \frac{1}{2}\left(1 - \frac{d}{2q}\right)$.

Local recovery of erased symbol at position $\mathbf{a} \in \mathbb{F}_q^2$

1. Pick a line $\ell = \{\mathbf{a} + \lambda \mathbf{b} \mid \lambda \in \mathbb{F}_q\}$ through a that has $\leqslant 2\tau q$ erasures ($\geqslant 1/2$ the lines are such)
2. Interpolate $f_\ell \in \mathbb{F}_q[\lambda]$ of degree d from values of f_ℓ and its derivative $\frac{df_\ell}{d\lambda}$ at the ($> d/2$) unerased points in ℓ.

Chain rule

$$\frac{df_\ell}{d\lambda}(\lambda_0) = \frac{\partial f}{\partial X_1}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_1 + \frac{\partial f}{\partial X_2}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_2$$

Now $f_\ell(0) = f(\mathbf{a})$, so value of f at $\mathbf{a}$ can be recovered.

But we also need to recover $\frac{\partial f}{\partial X_1}(\mathbf{a})$ and $\frac{\partial f}{\partial X_2}(\mathbf{a})$.

Directional derivatives

Chain rule: $\frac{df_\ell}{d\lambda}(\lambda_0) = \frac{\partial f}{\partial X_1}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_1 + \frac{\partial f}{\partial X_2}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_2$

Evaluating the derivative $\frac{df_\ell}{d\lambda}$ at $\lambda = 0$ gives the directional derivative of f at $\mathbf{a}$ in the direction $\mathbf{b}$ (which equals $\frac{\partial f}{\partial X_1}(\mathbf{a}) \cdot b_1 + \frac{\partial f}{\partial X_2}(\mathbf{a}) \cdot b_2$)

This gives partial information about the partials $\frac{\partial f}{\partial X_1}(\mathbf{a})$ and $\frac{\partial f}{\partial X_2}(\mathbf{a})$.

Directional derivatives

Chain rule: $\frac{df_\ell}{d\lambda}(\lambda_0) = \frac{\partial f}{\partial X_1}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_1 + \frac{\partial f}{\partial X_2}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_2$

Evaluating the derivative $\frac{df_\ell}{d\lambda}$ at $\lambda = 0$ gives the directional derivative of f at $\mathbf{a}$ in the direction $\mathbf{b}$ (which equals $\frac{\partial f}{\partial X_1}(\mathbf{a}) \cdot b_1 + \frac{\partial f}{\partial X_2}(\mathbf{a}) \cdot b_2$)

This gives partial information about the partials $\frac{\partial f}{\partial X_1}(\mathbf{a})$ and $\frac{\partial f}{\partial X_2}(\mathbf{a})$.

Final observation

If we compute directional derivative of f at a along another direction b' not collinear with b, then one can solve for $\frac{\partial f}{\partial X_1}(\mathbf{a})$ and $\frac{\partial f}{\partial X_2}(\mathbf{a})$.

Directional derivatives

Chain rule: $\frac{df_\ell}{d\lambda}(\lambda_0) = \frac{\partial f}{\partial X_1}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_1 + \frac{\partial f}{\partial X_2}(\mathbf{a} + \lambda_0 \mathbf{b}) \cdot b_2$

Evaluating the derivative $\frac{df_\ell}{d\lambda}$ at $\lambda = 0$ gives the directional derivative of f at $\mathbf{a}$ in the direction $\mathbf{b}$ (which equals $\frac{\partial f}{\partial X_1}(\mathbf{a}) \cdot b_1 + \frac{\partial f}{\partial X_2}(\mathbf{a}) \cdot b_2$)

This gives partial information about the partials $\frac{\partial f}{\partial X_1}(\mathbf{a})$ and $\frac{\partial f}{\partial X_2}(\mathbf{a})$.

Final observation

If we compute directional derivative of f at a along another direction b' not collinear with b, then one can solve for $\frac{\partial f}{\partial X_1}(\mathbf{a})$ and $\frac{\partial f}{\partial X_2}(\mathbf{a})$.

Locality

$\#$ queries $\leqslant 2q \leqslant O(\sqrt{n})$

Bivariate multiplicty 2 codes

LCCs with locality $r \leqslant O(n^{1/2})$ and rate $\to 2/3$.

Improving the parameters

Bivariate multiplicty 2 codes

LCCs with locality $r \leqslant O(n^{1/2})$ and rate $\to 2/3$.

Improving the parameters

- To improve locality, use m-variate polynomials for $m > 2$. Number of queries $\leqslant O(n^{1/m})$

Bivariate multiplicty 2 codes

LCCs with locality $r \leqslant O(n^{1/2})$ and rate $\to 2/3$.

Improving the parameters

- To improve locality, use m-variate polynomials for $m > 2$. Number of queries $\leqslant O(n^{1/m})$
- To improve rate, use higher order derivatives (all partials of order $< s$)

Bivariate multiplicty 2 codes

LCCs with locality $r \leqslant O(n^{1/2})$ and rate $\to 2/3$.

Improving the parameters

- To improve locality, use m-variate polynomials for $m > 2$. Number of queries $\leqslant O(n^{1/m})$
- To improve rate, use higher order derivatives (all partials of order $< s$)
- Letting m grow, and s grow even faster, we get n^{ε}-query LCCs with rate $1 - \gamma$ for any desired $\varepsilon, \gamma > 0$!

Theorem (High rate LCCs)

$\forall \varepsilon, \gamma > 0$, *there are rate* $(1 - \gamma)$ *(erasure) LCCs with locality* n^{ε}.

Local error-correction: Open problems

Surprising progress in constructions in recent years. But several big questions remain open:

- Do there exist LCCs with $O(1)$ locality and sub-exponential block length $n \leqslant \exp(o(k))$?
- How much smaller can we make locality of high rate (or constant rate) codes?

Local error-correction: Open problems

Surprising progress in constructions in recent years. But several big questions remain open:

- Do there exist LCCs with $O(1)$ locality and sub-exponential block length $n \leqslant \exp(o(k))$?
- How much smaller can we make locality of high rate (or constant rate) codes?
- Can one beat the encoding length achieved by matching vector based locally *decodable* codes?
- Can one improve any of the lower bounds on encoding length which are all *very* far from the upper bounds?

Thank you for your attention!

Questions?