Decoding Reed-Solomon codes by solving a bilinear system with a Gröbner basis approach

Magali Bardet, **Rocco Mora**, Jean-Pierre Tillich

April 6, 2021

Sorbonne Université, INRIA Paris

Overview

Reed-Solomon decoding problem

Correcting up to Sudan bound

Beyond Sudan bound

Experiments and conclusions

Reed-Solomon decoding problem

Reed-Solomon code

A **Reed-Solomon code** of length n and dimension k over $\mathbb{F}_q$ with support $\mathbf{a} = (a_i)_{1 \le i \le n} \in \mathbb{F}_q^n$ is

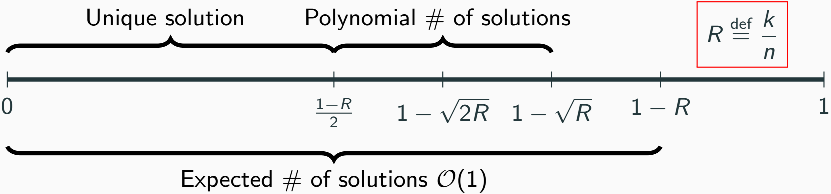
$$\mathbf{RS}_k(\mathbf{a}) = \{(P(a_i))_{1 \le i \le n} : \ P \in \mathbb{F}_q[X], \ \deg(P) < k\}.$$

Decoding problem

Given **a** and **b** (**received word**) and knowing that

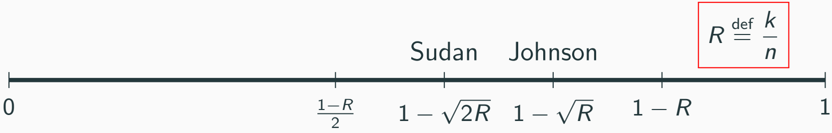
$$b_\ell = P(a_\ell) + e_\ell, \ \ \ell \in [\![1, n]\!],$$

with $t = \#\{i : e_i \neq 0\}$, retrieve P and e_ℓ, $\ \ell \in [\![1, n]\!]$.



Reed-Solomon decoding algorithms

(Beyond Berlekamp-Welch)



List decoding algorithms:

- Sudan '97: Sudan radius
- Guruswani, Sudan '98: Johnson radius

Power decoding algorithms:

- Schmidt, Sidorenko, Bossert '10: Sudan radius
- Nielsen '14: Sudan radius
- Nielsen '18: Johnson radius

Solving a bilinear system

Define

- $\Lambda(X) \stackrel{\text{def}}{=} \prod_{i:e_i \neq 0}(X - a_i) = X^t + \sum_{j=0}^{t-1} \lambda_j X^j$
  error locator polynomial,
- $P(X) = \sum_{i=0}^{k-1} p_i X^i$ corresponding to the codeword.

We can write n bilinear equations

$$\underbrace{P(a_\ell)}\underbrace{\Lambda(a_\ell)} = \underbrace{b_\ell}\underbrace{\Lambda(a_\ell)}, \quad \ell \in [\![1, n]\!]$$

- $P(a_\ell) = b_\ell, \quad e_\ell = 0$
- $\Lambda(a_\ell) = 0, \quad e_\ell \neq 0$

i.e.

$$\boxed{\sum_{i=0}^{k-1} \sum_{j=0}^{t} a_\ell^{i+j} p_i \lambda_j = \sum_{j=0}^{t} b_\ell a_\ell^j \lambda_j,} \quad \ell \in [\![1, n]\!] \text{ and } \lambda_t = 1.$$

Example: #errors$= d/2$

Parameters: $[n,k]_q = [9,3]_{31}$ RS code with $t = 3$ errors.

$p_0\lambda_0$	$p_1\lambda_0$	$p_2\lambda_0$	$p_0\lambda_1$	$p_1\lambda_1$	$p_2\lambda_1$	$p_0\lambda_2$	$p_1\lambda_2$	$p_2\lambda_2$	p_0	p_1	p_2	λ_0	λ_1	λ_2	1
1	8	2	8	2	16	2	16	4	16	4	1	13	11	26	22
1	15	8	15	8	27	8	27	2	27	2	30	9	11	10	26
1	30	1	30	1	30	1	30	1	30	1	30	2	29	2	29
1	27	16	27	16	29	16	29	8	29	8	30	18	21	9	26
1	17	10	17	10	15	10	15	7	15	7	26	24	5	23	19
1	28	9	28	9	4	9	4	19	4	19	5	9	4	19	5
1	5	25	5	25	1	25	1	5	1	5	25	8	9	14	8
1	26	25	26	25	30	25	30	5	30	5	6	27	20	24	4
1	3	9	3	9	27	9	27	19	27	19	26	4	12	5	15

Example: #errors$= d/2$

Parameters: $[n,k]_q = [9,3]_{31}$ RS code with $t = 3$ errors.

REDUCTION

$p_0\lambda_0$	$p_1\lambda_0$	$p_2\lambda_0$	$p_0\lambda_1$	$p_1\lambda_1$	$p_2\lambda_1$	$p_0\lambda_2$	$p_1\lambda_2$	$p_2\lambda_2$	p_0	p_1	p_2	λ_0	λ_1	λ_2	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12
0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	26
0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	2
0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	29
0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	7
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	28
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	24
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	29
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4

$$\Rightarrow \lambda_0 + 24 = 0, \quad \lambda_1 + 29 = 0, \quad \lambda_2 + 4 = 0.$$

Example: Sudan radius

Parameters: $[n, k]_q = [29, 5]_{61}$ RS code with $t = 15$ errors (Sudan bound).

	# equations	
	deg. 2	deg. 1
Bilinear system reduced	19	1+9
Multiply linear eq.s by p_i's and reduce	59	1+14
Multiply linear eq.s by p_i's and reduce	75	5+15=k+t

SOLVED

Macaulay matrix

The **Macaulay matrix** $\mathcal{M}_D^{\text{acaulay}}(\mathcal{S})$ in degree D of a set $\mathcal{S} = \{f_1, \cdots, f_m\}$ of polynomials is

$$\mathcal{M}_D^{\text{acaulay}}(\mathcal{S}) \stackrel{\text{def}}{=} \overset{\text{monomials of degree} \leq D}{\left(\qquad\qquad\qquad \right)} \leftarrow \begin{array}{c} x^\alpha f_i \text{ such that} \\ \deg(x^\alpha f_i) \leq D. \end{array}$$

Algorithm *D*-Gröbner Basis

Input

D Maximal degree,

$\mathcal{S}$ $= \{f_1, \cdots, f_m\}$ set of polynomials.

repeat

$\mathcal{S} \leftarrow \texttt{Pol}(\texttt{EchelonForm}(\mathcal{M}_D^{\text{acaulay}}(\mathcal{S})))$

until $\dim_{\mathbb{F}_q} \mathcal{S}$ has not increased.

Output $\mathcal{S}$.

Fact

When D is fixed, computing a D-Gröbner basis has polynomial complexity.

Fact

For large enough D, a D-Gröbner basis is a Gröbner basis (Lazard '83).

Monomial orders

Admissible monomial order

An **admissible monomial order** $<$ is an order on the monomials of $\mathbb{K}[x_1, \ldots, x_n]$ such that:

1. $<$ is total,
2. for any $m_1, m_2, m_3, \quad m_1 < m_2 \Rightarrow m_1 m_3 < m_2 m_3$
3. for any $m, \quad 1 < m$

Graded reverse lexicographic order (DRL) $x_1 > \cdots > x_n$

$$x^\alpha <_{\mathrm{drl}} x^\beta \iff \begin{array}{c} \deg\left(x^\alpha\right) < \deg\left(x^\beta\right) \\ \vee \\ \left(\deg\left(x^\alpha\right) = \deg\left(x^\beta\right)\right. \\ \left. \wedge\ \exists j \quad \text{s.t. } \left(\alpha_i = \beta_i, \quad \forall i > j\right) \wedge\ \alpha_j > \beta_j\right) \end{array}$$

Given $f = \sum_{\alpha \in \mathbb{N}^n} c_\alpha x^\alpha$,

- Leading Monomial: $LM(f) \stackrel{\text{def}}{=} \max_{c_\alpha \neq 0}(x^\alpha)$
- Leading Coefficient: $LC(f) \stackrel{\text{def}}{=} c_\alpha, \quad$ such that $LM(f) = x^\alpha$.
- Leading Term: $LT(f) \stackrel{\text{def}}{=} LC(f)LM(f)$.

Gröbner basis

Let $\mathcal{I}$ be an ideal of $\mathbb{K}[x_1, \ldots, x_n]$ and $<$ a monomial order. Then $G = \{g_1, \cdots, g_s\} \subset \mathcal{I}$ is a **Gröbner basis** of $\mathcal{I}$ if and only if

$$\langle LM(g_1), \cdots, LM(g_s) \rangle = \langle LM(f) : f \in \mathcal{I} \rangle.$$

Each ideal $\mathcal{I} \neq \{0\}$ admits a Gröbner basis (not unique).

Ideal Membership Problem

Given $f, g_1, \ldots, g_s \in \mathbb{K}[x_1, \ldots, x_n]$, detemine if $f \in \langle g_1, \ldots, g_s \rangle$.

Alternatively, determine if $\exists\ f_1, \ldots f_s \in \mathbb{K}[x_1, \ldots, x_n]$ s.t. $f = \sum_{i=1}^{s} f_i g_i$.

- Not trivial as in the univariate case
- Solved by Gröbner basis techniques

If $LM(g_i) \mid LM(f)$ then f can be **reduced** by g_i:

$$r \leftarrow f - \frac{LT(f)}{LT(g_i)} g_i$$

and $r = 0$ or $LM(r) < LM(f)$.

We can iterate this reduction until the remainder r is 0 or is no more divisible by any g_i.

Fact

If $G = \{g_1, \ldots, g_s\}$ is a Gröbner basis, then $f \in \mathcal{I}$ if and only if the last remainder r is 0.

Generalization of:

- Division in a univariate polynomial ring,
- Gaussian elimination.

Reduced Gröbner basis

Let G be a Gröbner basis for the ideal $\mathcal{I}$ wrt $<$. Then G is **reduced** if:

- $LC(g) = 1 \quad \forall g \in G$,
- For any $g \in G$, $\langle LT(G \setminus \{g\})\rangle$ does not contain any monomial of g.

Each ideal $\mathcal{I} \neq \{0\}$ admits a **unique** reduced Gröbner basis.

Consider the algebraic system
$$\begin{cases} f_1(x_1, \cdots, x_n) & = & 0 \\ \cdots & = & \\ f_m(x_1, \cdots, x_n) & = & 0 \end{cases}$$

Fact

If the system has a unique solution $(r_1, \cdots, r_n)$ and $\mathcal{I} = \langle f_1, \ldots, f_m\rangle$ is radical then the reduced Gröbner basis is given by $\{x_1 - r_1, \cdots, x_n - r_n\}$.

Here we are interested in graded orders.

Degree fall

A **degree fall** of degree s for $\mathcal{S} = \{f_1, \cdots, f_m\}$ is a polynomial combination $\sum_{i=1}^{m} g_i f_i$ which satisfies

$$0 < s \stackrel{\text{def}}{=} \deg \left(\sum_{i=1}^{m} g_i f_i \right) < \max_{i=1}^{m} \deg \left(g_i f_i \right).$$

If we are able to predict non-trivial degree falls we can speed up Gröbner basis computation.

Another description

- $R(X)$ interpolator polynomial (degree $\leq n - 1$)

$$R(a_\ell) = b_\ell, \quad \ell \in [\![1, n]\!]$$

- $G(X) \stackrel{\text{def}}{=} \Pi_{\ell=1}^{n}(X - a_\ell)$ (can be precomputed)

Key equation implicit in Gao's decoder

$$\Lambda(X)P(X) \equiv \Lambda(X)R(X) \mod G(X)$$

Proposition

$$\sum_{i=0}^{k-1} \sum_{j=0}^{t} a_\ell^{i+j} p_i \lambda_j = \sum_{j=0}^{t} b_\ell a_\ell^j \lambda_j, \quad \ell \in [\![1, n]\!]$$

and

$$\Lambda(X)P(X) \equiv \Lambda(X)R(X) \mod G(X)$$

are equivalent.

They can be obtained from each other by linear combinations.

Why do we use $\Lambda(X)P(X) \equiv \Lambda(X)R(X) \mod G(X)$?

- More convenient to work with to understand Gröbner basis calculations.
- They give directly $n-k-t+1$ linear equations, since
  - the coefficient of degree $d \in [\![t+k, n-1]\!]$ coincides with the coefficient of the same degree in $-R(X)\Lambda(X) \mod G(X)$ since $\Lambda(X)P(X)$ is of degree $\leq t+k-1$;
  - the coefficient of $S(X)$ of degree $t+k-1$ is equal to $p_{k-1} - \text{coeff}\left([\Lambda(X)R(X)]_{G(X)}, X^{t+k-1}\right)$ because $\Lambda(X)$ is monic and of degree t.

Correcting up to Sudan bound

Algorithm with $D = 2$ can decode up to Sudan bound

The Algorithm, with input the original bilinear system and $D = 2$, can decode up to Sudan decoding radius in polynomial time.

Symply powered key equations (Nielsen '14)

$$\Lambda(X)P(X)^u \equiv \Lambda(X)R(X)^u \mod G(X), \quad u \in \mathbb{Z}_+.$$

Proposition

Let $q_1 \stackrel{\text{def}}{=} \max\{u : t + (k-1)u \leq n-1\} = \left\lfloor \frac{n-t-1}{k-1} \right\rfloor$. All affine functions in the λ_i's of the form coeff $\left(\left[\Lambda(X) R^j(X) \right]_{G(X)}, X^u \right)$ for $j \in [\![1, q_1]\!]$ and $u \in [\![t + (k-1)j + 1, n-1]\!]$ are in the linear span of the set $\mathcal{S}$ output by the Algorithm with $D = 2$.

Proof (sketch)

- S contains the coefficients of

$$\Lambda(X)P(X) - \Lambda(X)R(X) \mod G(X)$$

  and therefore

$$\text{coeff}\left([-\Lambda(X)R(X)]_{G(X)}, X^u\right) \quad \text{for all } u \in [\![t+k, n-1]\!].$$

- By induction ($j \to j+1$):

$$\begin{aligned} &\left(\Lambda P^{j+1} - \Lambda R^{j+1}\right) \mod G \\ =& \left(P(\Lambda P^j - \Lambda R^j) + R^j(\Lambda P - \Lambda R)\right) \mod G \\ =& \left(P(\Lambda P^j - \Lambda R^j \mod G) + R^j(\Lambda P - \Lambda R \mod G)\right) \mod G. \end{aligned}$$

Split the sum:

- $$P(\Lambda P^j - \Lambda R^j \mod G) \mod G$$

- $$R^j(\Lambda P - \Lambda R \mod G) \mod G$$

Split the sum:

- $$P(\underbrace{\Lambda P^j - \Lambda R^j \mod G}_{\text{coefficients of degree in } [\![t+(k-1)j+1, n-1]\!] \text{ vanish}}) \mod G$$

- $$R^j(\Lambda P - \Lambda R \mod G) \mod G$$

Split the sum:

- $$\underbrace{P(\,\Lambda P^j - \Lambda R^j \mod G\,)}_{\text{polynomial of degree } \leq t + (k-1)(j+1) \text{ after elimination of variables}} \mod G$$

- $$R^j(\,\Lambda P - \Lambda R \mod G\,) \mod G$$

Split the sum:

- $$\underbrace{P(\Lambda P^j - \Lambda R^j \mod G) \mod G}_{\text{polynomial of degree } \leq t + (k-1)(j+1) \text{ after elimination of variables}}$$

- $$R^j(\Lambda P - \Lambda R \mod G) \mod G$$

Split the sum:

- $$\underbrace{P(\,\Lambda P^j - \Lambda R^j \mod G\,) \mod G}_{\text{polynomial of degree } \leq t+(k-1)(j+1) \text{ after elimination of variables}}$$

- $$R^j(\,\underbrace{\Lambda P - \Lambda R \mod G}_{\text{initial polynomial equations}}\,) \mod G$$

Split the sum:

- $$\underbrace{P(\Lambda P^j - \Lambda R^j \mod G) \mod G}_{\text{polynomial of degree } \leq t+(k-1)(j+1) \text{ after elimination of variables}}$$

- $$\underbrace{R^j(\Lambda P - \Lambda R \mod G) \mod G}_{\text{linear combination of equations in } S}$$

Split the sum:

- $$\underbrace{P(\,\Lambda P^j - \Lambda R^j \mod G\,) \mod G}_{\text{polynomial of degree } \leq t+(k-1)(j+1) \text{ after elimination of variables}}$$

- $$\underbrace{R^j(\,\Lambda P - \Lambda R \mod G\,) \mod G}_{\text{linear combination of equations in } S}$$

$\Rightarrow$ coeff $\left(\left[\Lambda(X)R^{j+1}(X)\right]_{G(X)}, X^u\right)$ are in the linear span of the set $\mathcal{S}$ output by a 2-Gröbner basis for $u \in [\![t+(k-1)(j+1)+1, n-1]\!]$.

Beyond Sudan bound

Example: above Sudan radius

Parameters: $[n, k]_q = [25, 5]_{31}$ RS code with $t = 15$ errors.

	# equations		
	deg. 3	deg. 2	deg. 1
Reduced matrix deg. 2		18	7
Multiply by p_i's and reduce	149	31	7
Multiply by λ_i's and reduce	262	38	7
Multiply by λ_i's and reduce	291	41	7
Multiply by λ_i's and reduce	297	50	7
Multiply by λ_i's and reduce	325	67	7
Multiply by λ_i's and reduce	335	91	$20 = t + k$

SOLVED

“Error evaluator” polynomial $\Omega(X)$

The “**error evaluator**” polynomial $\Omega(X)$ of degree $\leq t-1$ defined by

$$\Omega(a_\ell) = -e_\ell, \quad \text{for all } \ell \in [\![1, n]\!], \ e_\ell \neq 0.$$

We then have the identity

$$\Lambda(P - R) = \Omega G.$$

Equivalent definition of Ω as

$$\boxed{\Omega \stackrel{\text{def}}{=} -\Lambda R \div G.}$$

Fact

$\Omega(X)$'s coefficients are linear forms in the λ_i's.

Low-degree equations in the λ_i's

Generalization of Power decoding equations (Nielsen '18)

$$\Lambda^s P^u = \sum_{i=0}^{u} \left(\Lambda^{s-i}\Omega^i\right) \binom{u}{i} R^{u-i} G^i \qquad \stackrel{\text{def}}{=} \chi(s,u), \quad u \in [\![1, s-1]\!],$$

$$\Lambda^s P^u \equiv \left[\sum_{i=0}^{s-1} \left(\Lambda^{s-i}\Omega^i\right) \binom{u}{i} R^{u-i} G^i\right]_{G^s} \stackrel{\text{def}}{=} \chi(s,u), \quad u \in [\![s, v]\!].$$

From the identity

$$\left(\Lambda^s P^u\right)\left(\Lambda^{s'} P^{u'}\right) = \Lambda^{s+s'} P^{u+u'},$$

it is clear that

$$\boxed{\chi(s,u)\chi(s',u') - \chi(s+s', u+u') = 0}$$

Trivially produced at degree $s + s' + u + u'$ by a Gröbner basis, but actually discovered at a rather smaller degree.

- Let $\mathcal{I}_D = \langle \mathcal{S} \rangle_{\mathbb{F}_q}$ where $\mathcal{S}$ is the set output by the Algorithm with input D.
- $P \in_{\text{coef}} \mathcal{I}_v$ means that all the coefficients of P belong to $\mathcal{I}_v$.
- $\chi(s,u)_H \stackrel{\text{def}}{=} \sum_{i > ts+u(k-1)} a_i X^i$, where $\chi(s,u) = \sum_i a_i X^i$
- $q_s \stackrel{\text{def}}{=} \max\{u : st + u(k-1) \leq sn - 1\}$

Theorem

For all integers $1 \leq s$, $1 \leq s'$, $0 \leq u \leq q_s$, $0 \leq u' \leq q_{s'}$

$$\begin{aligned} \chi(s,u)_H &\in_{\text{coef}} \mathcal{I}_{s+1} \\ \chi(s,u)\chi(s',u') - \chi(s+s',u+u') &\in_{\text{coef}} \mathcal{I}_{s+s'+1}. \end{aligned}$$

Example ($s = s' = 1, u = 1, u' = 2$):

$$[\Lambda R]_G \cdot \left[\Lambda R^2\right]_G - \left[\Lambda^2 R^3 + 3\Lambda R^2 \Omega G\right]_{G^2} \in_{\text{coef}} \mathcal{I}_3.$$

Lemma

For all integers $1 \leq s$ *and* $0 \leq u < q_s$

$$\begin{aligned} \chi(s,u)P - \chi(s,u+1) &\in_{coef} \mathcal{I}_{s+1} \\ \chi(s,u+1)_H &\in_{coef} \mathcal{I}_{s+1}. \end{aligned}$$

Generalization of linear equations at degree 2 (Sudan bound).

- linear (in λ_i's) high coefficients $\rightarrow$ degree-s (in λ_i's) high coefficients,
- bilinear equations $\rightarrow$ equations of bidegree $(1, s)$.

Proof (sketch) of the Theorem

By induction (on u_1 and u_2).
Assume

$$\chi(s_1, u_1)\chi(s_2, u_2) - \chi(s_1 + s_2, u_1 + u_2) \in_{\text{coef}} \mathcal{I}_{s_1+s_2+1}.$$

The degree is $s_1 + s_2$, therefore

$$P\chi(s_1, u_1)\ \chi(s_2, u_2) - \ P\chi(s_1 + s_2, u_1 + u_2)\ \ \in_{\text{coef}} \mathcal{I}_{s_1+s_2+1}.$$

By the previous Lemma,

$$\chi(s_1, u_1 + 1)\ \chi(s_2, u_2) - \ \chi(s_1 + s_2, u_1 + u_2 + 1)\ \ \in_{\text{coef}} \mathcal{I}_{s_1+s_2+1}.$$

Proof (sketch) of the Theorem

By induction (on u_1 and u_2).
Assume

$$\chi(s_1, u_1)\chi(s_2, u_2) - \chi(s_1 + s_2, u_1 + u_2) \in_{\text{coef}} \mathcal{I}_{s_1+s_2+1}.$$

The degree is $s_1 + s_2$, therefore

$$\boxed{P\chi(s_1, u_1)}\,\chi(s_2, u_2) - \; P\chi(s_1 + s_2, u_1 + u_2) \; \in_{\text{coef}} \mathcal{I}_{s_1+s_2+1}.$$

By the previous Lemma,

$$\boxed{\chi(s_1, u_1 + 1)}\,\chi(s_2, u_2) - \; \chi(s_1 + s_2, u_1 + u_2 + 1) \; \in_{\text{coef}} \mathcal{I}_{s_1+s_2+1}.$$

Proof (sketch) of the Theorem

By induction (on u_1 and u_2).
Assume

$$\chi(s_1, u_1)\chi(s_2, u_2) - \chi(s_1 + s_2, u_1 + u_2) \in_{\mathrm{coef}} \mathcal{I}_{s_1+s_2+1}.$$

The degree is $s_1 + s_2$, therefore

$$\boxed{P\chi(s_1, u_1)}\,\chi(s_2, u_2) - \boxed{P\chi(s_1 + s_2, u_1 + u_2)} \in_{\mathrm{coef}} \mathcal{I}_{s_1+s_2+1}.$$

By the previous Lemma,

$$\boxed{\chi(s_1, u_1 + 1)}\,\chi(s_2, u_2) - \boxed{\chi(s_1 + s_2, u_1 + u_2 + 1)} \in_{\mathrm{coef}} \mathcal{I}_{s_1+s_2+1}.$$

An alternative approach for Gröbner basis computation

1. Compute the polynomials in only λ_i's from the theorem
2. Run the Algorithm with maximal degree D of the system generated in this way
3. Recover the p_i's by solving a linear system once the λ_i's have been retrieved.

Experiments and conclusions

Experimental results

For some parameters, quadratic equations involving only λ_i's are enough to solve the system, and we don't need to go to degree 3 (unlike the bilinear system).

Table 1: $[n, k]_q = [64, 27]_{64}$

t	$\#\lambda_j$	Eq.	#Eq.	D	Max Matrix	$\mathbb{C}$
20	3	Bilinear system	2:46	3	1522×1800	$2^{26.5}$
		System in λ_i's	2:9	2	47×28	$2^{24.4}$

Table 2: $[n, k]_q = [256, 63]_{256}$

t	$\#\lambda_j$	Eq.	#Eq.	D	Max Matrix	$\mathbb{C}$
120	36	Bilinear system	2:182	3	20023×128018	$2^{38.0}$
		System in λ_i's	2:85	2	119×703	$2^{34.5}$

When the number of remaining λ_j's is small compared to the number of p_i's, even if the maximal degree D is larger than for the bilinear system, the number of variables is much smaller and the computation is faster.

Table 3: $[n,k]_q = [64,27]_{64}$

t	$\#\lambda_j$	Eq.	#Eq.	D	Max Matrix	$\mathbb{C}$
23	9	Bilinear system	2:49	5	428533×406773	$2^{45.4}$
		System in λ_i's	2:4, 3:22	5	1466×1641	$2^{30.1}$
24	11	Bilinear system	2:50	≥ 6	–	–
		System in λ_i's	2:1, 3:23	7	28199×23536	$2^{35.8}$

Table 4: $[n,k]_q = [256,63]_{256}$

t	$\#\lambda_j$	Eq.	#Eq.	D	Max Matrix	$\mathbb{C}$
124	48	Bilinear system	2:186	≥ 4	–	–
		System in λ_i's	2:117, 3:1, 4:189	4	164600×270725	$2^{45.2}$

In some cases we can efficiently attain and even slightly pass Johnson bound.

Table 5: $[n, k]_q = [64, 27]_{64}$

t	$\#\lambda_j$	Eq.	#Eq.	D	Max Matrix	$\mathbb{C}$
23 (JB)	9	Bilinear system	2:49	5	428533×406773	$2^{45.4}$
		System in λ_i's	2:4, 3:22	5	1466×1641	$2^{30.1}$
24	11	Bilinear system	2:50	≥ 6	–	–
		System in λ_i's	2:1, 3:23	7	28199×23536	$2^{35.8}$

Table 6: $[n, k]_q = [37, 5]_{61}$

t	$\#\lambda_j$	Eq.	#Eq.	D	Max Matrix	$\mathbb{C}$
24 (JB)	12	Bilinear system	2:28	3	1065×1034	$2^{26.0}$
		System in λ_i's	2:37	3	454×454	$2^{28.0}$
25	15	Bilinear system	2:29	3	2520×1573	$2^{28.0}$
		System in λ_i's	2:25, 3:40	4	3193×3311	$2^{34.3}$
26	18	Bilinear system	2:30	4	20446×15171	$2^{33.1}$
		System in λ_i's	2:25, 3:37, 4:37	5	38796×22263	$2^{38.1}$
27	21	Bilinear system	2:31	4	27366×24894	$2^{36.0}$

Conclusions

- We proved that Gröbner bases can solve in polynomial time the bilinear system associated to the decoding problem of Reed-Solomon codes up to Sudan bound.
- We started to figure out why this Gröbner basis approach behaves much better here than for a random bilinear system (by predicting some unusual degree falls that may determine other degree falls).
- We proposed an alternative polynomial system to work with and showed that this is in some cases more convenient than taking the original bilinear system.
- We experimentally found several regions of parameters for which the Gröbner basis approach can decode efficiently up to and slightly beyond Johnson bound.

Thank you for your attention!

Questions?