Current State of the Art of the General Rank Decoding Problem

Anna-Lena Horlemann-Trautmann

University of St. Gallen, Switzerland

January 16th, 2019
2nd CWC in Barranquilla

The general (unique) rank decoding problem

Our notion of rank of $e \in \mathbb{F}_{q^m}^n$:

$$\text{rank}(e) = \max \# \text{ of } \mathbb{F}_q\text{-linearly independent coordinates of } e.$$

The problem

Consider a $\mathbb{F}_{q^m}$-linear code $C \subseteq \mathbb{F}_{q^m}^n$ with rank error correction capability t. Given a received word $r = c + e$ with $c \in C$ and $e \in \mathbb{F}_{q^m}^n$ with $\text{rank}(e) \leq t$, find the (unique) closest codeword c.

Of course, we can always do this by brute force, but the question is how to do so *efficiently*.

1. Why do we care? – Code based cryptography!

2. General rank (syndrome) decoding algorithms
   - with error spaces
   - with Grassmann support
   - with linearized polynomials

3. Conclusion and outlook

Post-quantum cryptography

- Most asymmetric cryptosystems in use are based on one of the following hard problems:
  - integer factorization
  - discrete logarithm
  - elliptic curve discrete logarithm
- These problems are not "hard enough" anymore on quantum computers, due to Shor's algorithm (1994).
- This calls for different cryptosystems, based on other hard mathematical problems. This is the field of *post-quantum cryptography*.

Post-quantum cryptography

The most studied proposed hard problems are

- the general syndrome decoding problem (code-based cryptography)
- the lattice shortest vector problem (lattice-based cryptography)
- inverting hash functions (hash-based cryptography)
- solving systems of multivariate polynomial equations (multivariate cryptography)
- walks in a supersingular isogeny graph (supersingular elliptic curve isogeny cryptography)

Code-based cryptography

- Two general schemes for public key cryptosystems: McEliece and Niederreiter.
- Both are equivalent from a security point of view.
- McEliece is more intuitive for coding theorists and is hence studied more often, Niederreiter is more efficient in implementation.

Code-based cryptography

- Two general schemes for public key cryptosystems: McEliece and Niederreiter.
- Both are equivalent from a security point of view.
- McEliece is more intuitive for coding theorists and is hence studied more often, Niederreiter is more efficient in implementation.
- The original McEliece proposal uses binary Goppa codes and has been unbroken for 30 years.
- Advantage: Computations are fast.
- Disadvantage: Public key size is a lot larger than in other systems.

Main idea of code-based cryptosystems

- Decoding a random linear code is a hard problem,
- but decoding codes with structure can often be done very efficiently.
- Private key is some code with an efficient decoding algorithm.
- Public key is a disguised version of this code, looking like a random code.

McEliece cryptosystem

Let ϕ be some disguising function such that the weight of any vector can decrease at most by $2\hat{t}$.

1. **Private key**: a generator matrix G_{priv} of a code C with a known efficient decoding algorithm (and err. corr. cap. t)
2. **Public key**: the disguised generator matrix $G_{pub} = \phi(G_{priv})$ and the error correction capability $t - \hat{t}$
3. **Encryption:** message m is encrypted as

$$\hat{m} = mG_{pub} + e$$

   where e is an error of weight at most $t - \hat{t}$
4. **Decryption:**

$$m = \text{decode}(\phi^{-1}(\hat{m}))$$

Crucial points

- We need a code C that has efficient encoding and decoding algorithms.
- We assume that the code family used is known, hence we require this family to have enough elements to prevent brute force attacks.
- The error correction capability needs to be large enough such that a brute force attack on the possible errors can be prevented.
- We need a disguising function such that the original code cannot be found from the public generator matrix.
- Any generic decoding (in part. information set decoding) should be infeasible in the public code.
- The size of the generator matrix is the key size, which we want to have as small as possible.

Reed-Solomon codes

Good:

- Efficient encoding and decoding algorithms.
- Code family is large enough.
- Error correction capability is as large as possible.

Bad:

- Key size is large.
- Distinguisher attack possible!

Binary Goppa codes

Good:

- Efficient encoding and decoding algorithms.
- Code family is large enough.
- Error correction capability is good.
- No efficient distinguisher attack known.

Bad:

- Key size is very large.

Key size in bits (approximate, in 2010)

bit security	80	96	112
key size RSA	1024	...	2048
key size Goppa-McEliece	588777	1056751	1548288

Key size in bits (approximate, in 2010)

bit security	80	96	112
key size RSA	1024	...	2048
key size Goppa-McEliece	588777	1056751	1548288

Idea to reduce key size: Use rank metric instead of Hamming!

Why does the rank metric improve the key size?

- Assume we have no structural/distinguisher attacks.
- Key size depends on the complexity of the best known generic decoding algorithm – for linear codes a *syndrome decoding* algorithm.
- In the Hamming metric best algorithms are *information set decoding (ISD)* algorithms.
- In the rank metric known algorithms are much less efficient than in the Hamming metric).
  $\Longrightarrow$ smaller keys reach same security level

To determine the necessary key size, we need to know the complexity of solving the general rank decoding problem.

Key size in bits in applications (approximate)

bit security	128	192	256
RSA	3 072	7 680	15 360
Goppa-McEliece	$2 \cdot 10^6$	$4 \cdot 10^6$	$6 \cdot 10^6$
Gabidulin (DRANKULA)	62 000	118 160	216 000
QC LRPC (LOCKER)	5 893	8 383	9 523

Key size for almost GV-optimal codes (approximate)

bit security	128	256
Hamming metric	$100KB$	$350KB$
rank metric	$2.2KB$	$8.7KB$

1. Why do we care? – Code based cryptography!

2. General rank (syndrome) decoding algorithms
   - with error spaces
   - with Grassmann support
   - with linearized polynomials

3. Conclusion and outlook

Error space

- In the Hamming metric we often split the decoding process into first finding the support, and then the values, of the error vector.

Error space

- In the Hamming metric we often split the decoding process into first finding the support, and then the values, of the error vector.
- In the rank metric we can split into first finding the error space, and then the error vector.

Definition

Let $e \in \mathbb{F}_{q^m}^n$ be an error vector of rank t. Then

$$E = \langle e_1, \ldots, e_n \rangle_{\mathbb{F}_q}$$

is called the *error space* (or *support*) of e. It holds that $\dim_q(E) = t$.

Finding the error vector from the error space

- Let H be a parity check matrix and the syndrome

$$s = rH^\top = eH^\top.$$

- Assume you know the error space $E = \langle e_1, \ldots, e_n \rangle$ and a basis $E_1, \ldots, E_t$ of E.
- If we can solve the system of equations (exp. over $\mathbb{F}_q$)

$$(e_1, \ldots, e_n)H^\top = s$$

$$e_1 = \sum_{j=1}^{t} e_{1j} E_j$$

$$\vdots$$

$$e_n = \sum_{j=1}^{t} e_{nj} E_j$$

$e_{ij} \in \mathbb{F}_q$, then we find the error vector e.

Finding the error vector from the error space

- Expanded over $\mathbb{F}_q$, this system of equations has $m(n-k)$ equations and tn variables.
- If all equations linearly independent, necessary condition for unique solution:

$$tn \leq m(n-k)$$

(which is always true)

Finding the error vector from the error space

- Expanded over $\mathbb{F}_q$, this system of equations has $m(n-k)$ equations and tn variables.
- If all equations linearly independent, necessary condition for unique solution:

$$tn \leq m(n-k)$$

  (which is always true)
- Comparison: Without the knowledge of E we have nm variables, and we never have $mn \leq m(n-k)$.

Example

- Assume you have $q = 2^4$, $G = (1\ \alpha\ \alpha^2\ \alpha)$, $r = (\alpha + 1\ \alpha^2\ \alpha^3 + 1\ \alpha^2)$ and $E = \langle 1 \rangle = \mathbb{F}_2$.
- Compute

$$s = (\alpha + 1\ \alpha^2\ \alpha^3 + 1\ \alpha^2) \begin{pmatrix} \alpha & 1 & 0 & 0 \\ \alpha^2 & 0 & 1 & 0 \\ \alpha & 0 & 0 & 1 \end{pmatrix}^\top = (1\ 0\ 1).$$

Example

- Assume you have $q = 2^4$, $G = (1\ \alpha\ \alpha^2\ \alpha)$, $r = (\alpha + 1\ \alpha^2\ \alpha^3 + 1\ \alpha^2)$ and $E = \langle 1 \rangle = \mathbb{F}_2$.
- Compute

$$s = (\alpha + 1\ \alpha^2\ \alpha^3 + 1\ \alpha^2) \begin{pmatrix} \alpha & 1 & 0 & 0 \\ \alpha^2 & 0 & 1 & 0 \\ \alpha & 0 & 0 & 1 \end{pmatrix}^\top = (1\ 0\ 1).$$

- Solve $(e_1\ e_2\ e_3\ e_4)H^\top = (1\ 0\ 1)$ with $e_i \in \mathbb{F}_2$:

$$\begin{aligned} \alpha e_1 + e_2 = 1 &\implies e_1 = 0, e_2 = 1 \\ \alpha^2 e_1 + e_3 = 0 &\implies e_3 = 0 \\ \alpha e_1 + e_4 = 1 &\implies e_4 = 1 \end{aligned}$$

Example

- Assume you have $q = 2^4$, $G = (1\ \alpha\ \alpha^2\ \alpha)$, $r = (\alpha + 1\ \alpha^2\ \alpha^3 + 1\ \alpha^2)$ and $E = \langle 1 \rangle = \mathbb{F}_2$.
- Compute

$$s = (\alpha + 1\ \alpha^2\ \alpha^3 + 1\ \alpha^2) \begin{pmatrix} \alpha & 1 & 0 & 0 \\ \alpha^2 & 0 & 1 & 0 \\ \alpha & 0 & 0 & 1 \end{pmatrix}^\top = (1\ 0\ 1).$$

- Solve $(e_1\ e_2\ e_3\ e_4)H^\top = (1\ 0\ 1)$ with $e_i \in \mathbb{F}_2$:

$$\begin{aligned} \alpha e_1 + e_2 = 1 &\implies e_1 = 0, e_2 = 1 \\ \alpha^2 e_1 + e_3 = 0 &\implies e_3 = 0 \\ \alpha e_1 + e_4 = 1 &\implies e_4 = 1 \end{aligned}$$

- Solution: $e = (1\ 0\ 1\ 0)$ and $c = r - e = (\alpha\ \alpha^2\ \alpha^3\ \alpha^2)$

Algorithm 1 – syndrome decoding with guessing

Let $r = c + e$ and H be the parity check matrix

- Randomly choose an error space $E \subseteq \mathbb{F}_{q^m}$ of dimension t.
- Choose a basis $E_1, \ldots, E_t$ of E.
- Recovering the error vector e:
  Writing $e_i = \sum_{i=1}^{t} e_{ij} E_j$, solve the system $eH^\top = s$.
- If this is not possible, start over.

Algorithm 1 – syndrome decoding with guessing

Let $r = c + e$ and H be the parity check matrix

- Randomly choose an error space $E \subseteq \mathbb{F}_{q^m}$ of dimension t.
- Choose a basis $E_1, \ldots, E_t$ of E.
- Recovering the error vector e:
  Writing $e_i = \sum_{i=1}^{t} e_{ij} E_j$, solve the system $eH^\top = s$.
- If this is not possible, start over.

Problem: Too complex, since there are $\binom{m}{t}_q$ many possible E.

Information set decoding (ISD)

- The previous setup is analogous to the basic setup of ISD attacks in the Hamming metric, where we randomly choose information sets and check if the respective coordinates are the error support.

Information set decoding (ISD)

- The previous setup is analogous to the basic setup of ISD attacks in the Hamming metric, where we randomly choose information sets and check if the respective coordinates are the error support.
- The main improvement in good ISD attacks is how to cleverly choose the information sets.

Information set decoding (ISD)

- The previous setup is analogous to the basic setup of ISD attacks in the Hamming metric, where we randomly choose information sets and check if the respective coordinates are the error support.
- The main improvement in good ISD attacks is how to cleverly choose the information sets.
- In the rank metric this can be done e.g. by fixing a basis of some subspace of E and complementing it in various ways.
- References (e.g.): Chabaud-Stern ‘96, Ourivski-Johannson ‘02, Gaborit-Ruatta-Schrek(-Zemor) ‘14-‘16

Attention!

–

For some codes we can compute the error space.

Remaining problem: efficiently finding the error space

- For par. ch. matrix H, let $F_1, \ldots, F_d$ be basis for $\langle h_{ij} \rangle_{ij}$.
- Since $s_i = \sum_{j=1}^{n} e_j h_{ij}$ we have

$$s_i \in \langle E_1 F_1, E_1 F_2, \ldots, E_t F_d \rangle_{\mathbb{F}_q}$$

for all i, thus

$$S = \langle s_1, \ldots, s_{n-k} \rangle_{\mathbb{F}_q} \subseteq \langle E_1 F_1, E_1 F_2, \ldots, E_t F_d \rangle_{\mathbb{F}_q}.$$

- If we have

$$S = \langle E_1 F_1, E_1 F_2, \ldots, E_t F_d \rangle_{\mathbb{F}_q}$$

then $E \subseteq F_i^{-1} S$ for any i, and hence

$$E = F_1^{-1} S \cap F_2^{-1} S \cap \cdots \cap F_d^{-1} S.$$

Remaining problem: efficiently finding the error space

- For par. ch. matrix H, let $F_1, \ldots, F_d$ be basis for $\langle h_{ij} \rangle_{ij}$.
- Since $s_i = \sum_{j=1}^{n} e_j h_{ij}$ we have

$$s_i \in \langle E_1 F_1, E_1 F_2, \ldots, E_t F_d \rangle_{\mathbb{F}_q}$$

for all i, thus

$$S = \langle s_1, \ldots, s_{n-k} \rangle_{\mathbb{F}_q} \subseteq \langle E_1 F_1, E_1 F_2, \ldots, E_t F_d \rangle_{\mathbb{F}_q}.$$

- If we have

$$S = \langle E_1 F_1, E_1 F_2, \ldots, E_t F_d \rangle_{\mathbb{F}_q}$$

then $E \subseteq F_i^{-1} S$ for any i, and hence

$$E = F_1^{-1} S \cap F_2^{-1} S \cap \cdots \cap F_d^{-1} S.$$

- If we do not have equality in the sets, then the same procedure finds only a subspace of the error space.

When is $S = \langle EF \rangle$?

- If we assume that $\dim(\langle EF \rangle) = td$, then we need

$$td \stackrel{!}{=} \dim(S) \leq n - k.$$

$\implies$ We need low d!
(This is the idea of low rank parity check codes.)

When is $S = \langle EF \rangle$?

- If we assume that $\dim(\langle EF \rangle) = td$, then we need

$$td \stackrel{!}{=} \dim(S) \leq n - k.$$

$\implies$ We need low d!
(This is the idea of low rank parity check codes.)

- For $d = 2$ we can (possibly) correct $t \leq (n-k)/2$ errors.

When is $S = \langle EF \rangle$?

- If we assume that $\dim(\langle EF \rangle) = td$, then we need

$$td \stackrel{!}{=} \dim(S) \leq n - k.$$

$\implies$ We need low d!
(This is the idea of low rank parity check codes.)

- For $d = 2$ we can (possibly) correct $t \leq (n-k)/2$ errors.
- If we have $\dim(S) < n - k$, we can do probabilistic decoding with error failure in $q^{-(n-k-td)}$.

Algorithm 2 – syndrome decoding (for LRPC)

Let $r = c + e$ and H be the parity check matrix. $F_1, \ldots, F_d$ is basis of $\langle h_{ij} \rangle_{ij}$.

- Syndrome space computation:

$$(s_1, \ldots, s_{n-k}) = rH^\top$$

$$S = \langle s_1, \ldots, s_{n-k} \rangle$$

- Recovering the error space E:

$$S_i = F_i^{-1} S$$

$$E = S_1 \cap S_2 \cap \cdots \cap S_d.$$

  $E_1, \ldots, E_t$ is basis of E.

- Recovering the error vector e:
  Writing $e_i = \sum_{i=1}^{t} e_{ij} E_j$, solve the system $eH^\top = s$.

References: Gaborit et al. '13, Aragon et al. '18

Example

- Assume you have $q = 2^4$, $G = (1\ \alpha\ \alpha\ \alpha)$, and $r = (\alpha + 1\ \alpha^2\ \alpha^2 + 1\ \alpha^2)$.
- Compute

$$s = (\alpha + 1\ \alpha^2\ \alpha^2 + 1\ \alpha^2) \begin{pmatrix} \alpha & 1 & 0 & 0 \\ \alpha & 0 & 1 & 0 \\ \alpha & 0 & 0 & 1 \end{pmatrix}^\top = (\alpha\ \alpha + 1\ 1)$$

  and thus $S = \langle 1, \alpha \rangle$.
- Basis for $\langle h_{ij} \rangle$ is $F_1 = 1, F_2 = \alpha$.
- Compute

$$E = F_1^{-1} S \cap F_2^{-1} S = \langle 1, \alpha^{-1} \rangle \cap \langle 1, \alpha \rangle = \langle 1 \rangle.$$

- Then solving syndrome equation as before gives

$$e = (1\ 0\ 1\ 0).$$

A different notion of rank support

–

the Grassmann support

ISD variants with Grassmann support

- All the previously described algorithms first find the error space E.
- We can decompose

$$(e_1, \ldots, e_n) = \underbrace{(E_1, \ldots, E_t)}_{\in \mathbb{F}_{q^m}^t} \underbrace{U}_{\in \mathbb{F}_q^{t \times n}}$$

  where $E_1, \ldots, E_t$ is a basis of E. We call $\mathrm{rs}(U)$ the *Grassmann support* of e.
- We can alter all algorithms to smartly guess U (instead of $E_1, \ldots, E_t$) first, and then solve the syndrome equation.
- References (e.g.): Ourivski-Johannson '02, current research

ISD variants with Grassmann support

- Now we need to guess a t-dimensional space in $\mathbb{F}_q^n$, instead of $\mathbb{F}_q^m$.
- This gives an improvement in the attack if

$$n < m.$$

- Open question: Is this the only way to do ISD attacks in the rank metric? Can we use another notion of support and hence information sets? Can we do other ways of "splitting" the problem?

Yet another approach

–

finding the Grassmann support with $\mathbb{F}_q$-subcodes

Decoding with the Grassmann support

- Remember that $e \in \mathbb{F}_{q^m}^n$ of rank t can be decomposed as $e = vU$ with $v \in \mathbb{F}_{q^m}^t$ or rank t and $U \in \mathbb{F}_q^{t \times n}$.

Decoding with the Grassmann support

- Remember that $e \in \mathbb{F}_{q^m}^n$ of rank t can be decomposed as $e = vU$ with $v \in \mathbb{F}_{q^m}^t$ or rank t and $U \in \mathbb{F}_q^{t \times n}$.
- Once we know U, we can compute the parity check matrix H_U and solve

$$\underbrace{r}_{c+vU} H_U^\top = \underbrace{xG}_{c} H_U^\top$$

(or equivalently solve the syndrome equation).

Decoding with the Grassmann support

- Remember that $e \in \mathbb{F}_{q^m}^n$ of rank t can be decomposed as $e = vU$ with $v \in \mathbb{F}_{q^m}^t$ or rank t and $U \in \mathbb{F}_q^{t \times n}$.
- Once we know U, we can compute the parity check matrix H_U and solve

$$\underbrace{r}_{c+vU} H_U^\top = \underbrace{xG}_{c} H_U^\top$$

(or equivalently solve the syndrome equation).

Theorem (HT-Marshall-Rosenthal '16)

Let $e \in \mathbb{F}_{q^m}^n$ be of rank t, s be such that $\gcd(s, m) = 1$, and let $\mathcal{S} \subseteq \mathbb{F}_{q^m}^n$ with $e \in \mathcal{S}$. Then,

$$\mathrm{supp}_{Gr}(e) \subseteq \sum_{i=0}^{r-1} \mathcal{S}^{(q^{si})}.$$

Decoding with the Grassmann support

- Remember that $e \in \mathbb{F}_{q^m}^n$ of rank t can be decomposed as $e = vU$ with $v \in \mathbb{F}_{q^m}^t$ or rank t and $U \in \mathbb{F}_q^{t \times n}$.
- Once we know U, we can compute the parity check matrix H_U and solve

$$\underbrace{r}_{c+vU} H_U^\top = \underbrace{xG}_{c} H_U^\top$$

(or equivalently solve the syndrome equation).

Theorem (HT-Marshall-Rosenthal '16)

Let $e \in \mathbb{F}_{q^m}^n$ be of rank t, s be such that $\gcd(s, m) = 1$, and let $\mathcal{S} \subseteq \mathbb{F}_{q^m}^n$ with $e \in \mathcal{S}$. Then,

$$\mathrm{supp}_{Gr}(e) \subseteq \sum_{i=0}^{r-1} \mathcal{S}^{(q^{si})}.$$

We can choose $\mathcal{S} = \langle G, r \rangle$, then $e \in \mathcal{S}$.

Some preliminaries

- We have $\mathrm{supp}_{Gr}(e) \subseteq \sum_{i=0}^{r-1} \mathcal{S}^{(q^{si})}$ and $\mathrm{supp}_{Gr}(e) \subseteq \mathbb{F}_q^n$

$$\implies \mathrm{supp}_{Gr}(e) \subseteq \sum_{i=0}^{r-1} \mathcal{S}^{(q^{si})} \cap \mathbb{F}_q^n.$$

- We can find the $\mathbb{F}_q$-subspace of any code C with generator matrix G in RREF by solving

$$\sum_{i=0}^{k-1} a_i (G_i^{([1])} - G_i) = \mathbf{0},$$

where $a_i \in \mathbb{F}_q$.

Algorithm 3 – Grassmann support decoding

- Construct the matrix

$$G_{\text{ext}} = \begin{pmatrix} G \\ \boldsymbol{r} \\ \vdots \\ G^{(q^{t-1})} \\ \boldsymbol{r}^{(q^{t-1})} \end{pmatrix}.$$

- Compute the space $\mathcal{U}$ generated by the $\mathbb{F}_q$-elements in $\mathcal{C}_{\text{ext}} = \langle G_{\text{ext}} \rangle_{\mathbb{F}_{q^m}}$.
- Compute a parity check matrix $H_U \in \mathbb{F}_q^{(n-u)\times n}$ for $\mathcal{U}$.
- Solve $rH_U^\top = x(GH_U^\top)$ for x.
- Reference: HT-Marshall-Rosenthal '16

Example

- Assume you have $q = 2^4$, $G = (1\ \alpha\ \alpha^2\ \alpha)$, $r = (0\ \alpha^2\ \alpha^3 + \alpha\ \alpha^2)$.
- $$G_{\text{ext}} = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha \\ 0 & \alpha^2 & \alpha^3 + \alpha & \alpha^2 \end{pmatrix} \to \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & \alpha^3 + \alpha & 1 \end{pmatrix}$$
- The $\mathbb{F}_2$-elements of this code are zero and $U = (1\ 0\ 1\ 0)$.

Example

- Assume you have $q = 2^4$, $G = (1\ \alpha\ \alpha^2\ \alpha)$, $r = (0\ \alpha^2\ \alpha^3 + \alpha\ \alpha^2)$.
- $$G_{\text{ext}} = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha \\ 0 & \alpha^2 & \alpha^3 + \alpha & \alpha^2 \end{pmatrix} \to \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & \alpha^3 + \alpha & 1 \end{pmatrix}$$
- The $\mathbb{F}_2$-elements of this code are zero and $U = (1\ 0\ 1\ 0)$.
- Compute
$$H_U = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$
and $GH_U = (\alpha\ \alpha^2 + 1\ \alpha)$.

Example

- Assume you have $q = 2^4$, $G = (1\ \alpha\ \alpha^2\ \alpha)$, $r = (0\ \alpha^2\ \alpha^3 + \alpha\ \alpha^2)$.
- $$G_{\text{ext}} = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha \\ 0 & \alpha^2 & \alpha^3 + \alpha & \alpha^2 \end{pmatrix} \to \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & \alpha^3 + \alpha & 1 \end{pmatrix}$$
- The $\mathbb{F}_2$-elements of this code are zero and $U = (1\ 0\ 1\ 0)$.
- Compute
$$H_U = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$
and $GH_U = (\alpha\ \alpha^2 + 1\ \alpha)$.
- Solve $rH_U = xGH_U \iff (\alpha^2\ \alpha^3 + \alpha\ \alpha^2) = x(\alpha\ \alpha^2 + 1\ \alpha)$.

Example

- Assume you have $q = 2^4$, $G = (1\ \alpha\ \alpha^2\ \alpha)$, $r = (0\ \alpha^2\ \alpha^3 + \alpha\ \alpha^2)$.
- $$G_{\text{ext}} = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha \\ 0 & \alpha^2 & \alpha^3 + \alpha & \alpha^2 \end{pmatrix} \to \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & \alpha^3 + \alpha & 1 \end{pmatrix}$$
- The $\mathbb{F}_2$-elements of this code are zero and $U = (1\ 0\ 1\ 0)$.
- Compute
$$H_U = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$
and $GH_U = (\alpha\ \alpha^2 + 1\ \alpha)$.
- Solve $rH_U = xGH_U \iff (\alpha^2\ \alpha^3 + \alpha\ \alpha^2) = x(\alpha\ \alpha^2 + 1\ \alpha)$.
- Solution: $x = \alpha$, $e = xU = (\alpha\ 0\ \alpha\ 0)$ and $c = (\alpha\ \alpha^2\ \alpha^3\ \alpha^2)$.

When do we get a unique solution?

- We need that $GH_U^\top$ has full rank k.
- This is given if $C \cap \mathcal{U} = \{0\}$.
- We can upper bound the dimension of $\mathcal{U}$ as

$$\dim(\mathcal{U}) \leq \dim(\mathcal{C}_{\text{ext}}) \leq (k+1)t - (t-1)\ell$$

  where $\ell = \dim(C^{q^s} \cap C)$.
- We need $\dim(\mathcal{U}) \leq n - k$ to have $k \leq n - \dim(\mathcal{U})$.
- The above chain of inequalities is small, if ℓ is large; therefore, probably large ℓ works well.
- Also small k (low rate) is (probably) advantageous.
- However, these are not necessary conditions.

$\Longrightarrow$ Open question!

Now a (completely) different approach

–

linearized polynomials

Error space and linearized polynomials

- As before let $r = c + e$ with $\text{rank}(e) = t$.
- Then there exists a (monic) linearized polynomial of q-degree t

$$f(x) = \sum_{i=0}^{t} f_i x^{q^i}$$

with $f(e_i) = f(c_i - r_i) = 0$ for all i.

- When we rewrite $c = xG$, then solving the system

$$(\sum_{i=0}^{t} f_i(xG_1 - r_1)^{q^i}, \ldots, \sum_{i=0}^{t} f_i(xG_n - r_n)^{q^i}) = (0, \ldots, 0)$$

would recover x (and equivalently c).

Reference: Gaborit et al. ‘16

When is this system solvable?

- This system has n equations and $k+t$ variables. Hence, we require

$$k + t \leq n,$$

which is always fulfilled.

When is this system solvable?

- This system has n equations and $k+t$ variables. Hence, we require
$$k + t \leq n,$$
which is always fulfilled.
- Problem: High degree polynomial equations, difficult to solve.
- Methods: Gröbner bases, linearization.

When is this system solvable?

- This system has n equations and $k+t$ variables. Hence, we require
$$k+t \leq n,$$
which is always fulfilled.
- Problem: High degree polynomial equations, difficult to solve.
- Methods: Gröbner bases, linearization.
- **Algorithm 4:** If we view the products $f_i x_j$ as separate new variables, then we get a system of linear equations with $k+t+kt$ variables.
- In this case we need
$$k+t+kt = (k+1)(t+1)-1 \leq n,$$
which is a strong restriction.

1. Why do we care? – Code based cryptography!

2. General rank (syndrome) decoding algorithms
   - with error spaces
   - with Grassmann support
   - with linearized polynomials

3. Conclusion and outlook

Conclusion

- The general decoding problem is particularly important for code based cryptography.
- In the Hamming metric information set decoding (ISD) is the most efficient for a random code.
- In the rank metric ISD is a lot less efficient.
  $\Longrightarrow$ Lower key size for same security level!
- Careful: other efficient decoding algorithms exist, if the random (public) code has certain properties.

Outlook – What about other metrics?

- Subspace metric – no good attack known, but also no good encryption possible.
- Lee metric – ISD can be adapted to $\mathbb{Z}_m$, but is less efficient than over $\mathbb{F}_q$. (Weger - Horlemann)
- Euclidean metric – this is basically lattice-based cryptography (closest vector problem $\iff$ shortest vector problem).

Outlook – What about other metrics?

- Subspace metric – no good attack known, but also no good encryption possible.
- Lee metric – ISD can be adapted to $\mathbb{Z}_m$, but is less efficient than over $\mathbb{F}_q$. (Weger - Horlemann)
- Euclidean metric – this is basically lattice-based cryptography (closest vector problem $\iff$ shortest vector problem).

Thank you for your attention!
Questions? – Comments?