# Gabidulin Decoding via Minimal Bases of Linearized Polynomial Modules

Anna-Lena Trautmann

Department of Electrical & Electronic Engineering, University of Melbourne
Department of Electrical & Computer Systems Eng., Monash University

January 28th, 2015
BIRS, Banff

In collaboration with Margreta Kuijper.

## What's in the title?

*Gabidulin codes* (Gabidulin 1985, Delsarte 1978) are useful in e.g. (random) linear network coding, space-time coding, crisscross error correction, distributed storage.

## What's in the title?

*Gabidulin codes* (Gabidulin 1985, Delsarte 1978) are useful in e.g. (random) linear network coding, space-time coding, crisscross error correction, distributed storage.

These codes can be defined with the help of *linearized polynomials*.

## What's in the title?

*Gabidulin codes* (Gabidulin 1985, Delsarte 1978) are useful in e.g. (random) linear network coding, space-time coding, crisscross error correction, distributed storage.

These codes can be defined with the help of *linearized polynomials*.

We use a *module* set up over the ring of linearized polynomials to decode Gabidulin codes within any radius (unique or list decoding).

## What's in the title?

*Gabidulin codes* (Gabidulin 1985, Delsarte 1978) are useful in e.g. (random) linear network coding, space-time coding, crisscross error correction, distributed storage.

These codes can be defined with the help of *linearized polynomials*.

We use a *module* set up over the ring of linearized polynomials to decode Gabidulin codes within any radius (unique or list decoding).

For our parametrization of all closest codewords to the received word we need *minimal bases* of the modules.

## Related Work

- Unique minimum distance decoding of Gabidulin codes: Gabidulin (1985,1992), Loidreau (2006), Richter-Plass (2004), Silva-Kschischang (2009-2011), Wachter-Zeh (2013), ...
- List-decoding of special types of Gabidulin codes and subcodes: Loidreau (2006), Mahdavifar-Vardy (2012), Wachter-Zeh (2013), Guruswami-Wang (2014), ...
- List-decoding lifted Gabidulin codes: Xie-Yan-Suter (2011), T'-Silberstein-Rosenthal (2013), ...

## Compared to our Work

- We use interpolation set up given for unique decoding by Loidreau.

## Compared to our Work

- We use interpolation set up given for unique decoding by Loidreau.
- Like the Gao-type unique decoder from Wachter-Zeh we use the Euclidean algorithm.

## Compared to our Work

- We use interpolation set up given for unique decoding by Loidreau.
- Like the Gao-type unique decoder from Wachter-Zeh we use the Euclidean algorithm.
- NEW: We use a module set up and a parametrization that allows us to find either
  - ALL closest codewords, or
  - COMPLETE list of codewords within ANY given radius

  (with better complexity than exhaustive search, especially for high rate codes).

## Compared to our Work

- We use interpolation set up given for unique decoding by Loidreau.
- Like the Gao-type unique decoder from Wachter-Zeh we use the Euclidean algorithm.
- NEW: We use a module set up and a parametrization that allows us to find either
  - ALL closest codewords, or
  - COMPLETE list of codewords within ANY given radius

  (with better complexity than exhaustive search, especially for high rate codes).
- This parametrization is analogous to the one for RS-codes by Ali-Kuijper (2011), but now for modules over rings of linearized polynomials equipped with composition (instead of multiplicative polynomial rings).

## Definition

A *q-linearized polynomial* is of the form $f(x) = \sum_{i=0}^{n} a_i x^{q^i}$ for $a_i \in \mathbb{F}_{q^m}$. If $a_n \neq 0$, $n$ is called the *q-degree* of $f(x)$.

## Definition

A *q-linearized polynomial* is of the form $f(x) = \sum_{i=0}^{n} a_i x^{q^i}$ for $a_i \in \mathbb{F}_{q^m}$. If $a_n \neq 0$, $n$ is called the *q-degree* of $f(x)$.

## Lemma

The set $\mathcal{L}_q(x, q^m)$ of all $q$-linearized polynomials over $\mathbb{F}_{q^m}$ forms a non-commutative ring, equipped with the normal addition $+$ and composition (symbolic multiplication) $\circ$.

## Definition

A *q-linearized polynomial* is of the form $f(x) = \sum_{i=0}^{n} a_i x^{q^i}$ for $a_i \in \mathbb{F}_{q^m}$. If $a_n \neq 0$, $n$ is called the *q-degree* of $f(x)$.

## Lemma

The set $\mathcal{L}_q(x, q^m)$ of all $q$-linearized polynomials over $\mathbb{F}_{q^m}$ forms a non-commutative ring, equipped with the normal addition $+$ and composition (symbolic multiplication) $\circ$.

*Symbolic division*: $f(x)$ is symbolically divisible on the left by $g(x)$ with quotient $m(x)$ if $g(m(x)) = f(x)$

## Definition

Let $g_1, \ldots, g_n \in \mathbb{F}_{q^m}$ be linearly independent over $\mathbb{F}_q$. Then

$$C_{Gab} := \{(f(g_1), \ldots, f(g_n)) \mid f(x) \in \mathcal{L}_q(x, q^m), \mathrm{qdeg} < k\}$$

is called a *Gabidulin code* in $\mathbb{F}_{q^m}^n$ of dimension $k$.

## Definition

Let $g_1, \ldots, g_n \in \mathbb{F}_{q^m}$ be linearly independent over $\mathbb{F}_q$. Then

$$C_{Gab} := \{(f(g_1), \ldots, f(g_n)) \mid f(x) \in \mathcal{L}_q(x, q^m), \mathrm{qdeg} < k\}$$

is called a *Gabidulin code* in $\mathbb{F}_{q^m}^n$ of dimension $k$.

Elements in $\mathbb{F}_{q^m}^n$ can be represented in $\mathbb{F}_q^{m \times n}$.

## Definition

Rank-Metric:

$$d_R(A, B) = \mathrm{rank}_q(A - B)$$

## Definition

Let $g_1, \ldots, g_n \in \mathbb{F}_{q^m}$ be linearly independent over $\mathbb{F}_q$. Then

$$C_{Gab} := \{(f(g_1), \ldots, f(g_n)) \mid f(x) \in \mathcal{L}_q(x, q^m), \operatorname{qdeg} < k\}$$

is called a *Gabidulin code* in $\mathbb{F}_{q^m}^n$ of dimension $k$.

Elements in $\mathbb{F}_{q^m}^n$ can be represented in $\mathbb{F}_q^{m \times n}$.

## Definition

Rank-Metric:

$$d_R(A, B) = \operatorname{rank}_q(A - B)$$

## Theorem

A Gabidulin code $C \subseteq \mathbb{F}_{q^m}^n$ of dimension $k$ has minimum rank distance $n - k + 1$, which is optimal (Singleton bound).
Therefore, it is called a *maximum rank distance (MRD) code*.

**Example:**
Consider $\mathbb{F}_{2^2} \cong \mathbb{F}_2[\alpha]$, with $\alpha^2 + \alpha + 1 = 0$. Fix $g_1 = 1, g_2 = \alpha$ (linearly independent). The Gabidulin code of length $n = 2$, dimension $k = 1$ and minimum distance $n - k + 1 = 2$ is

$$(0 \quad 0) \mapsto \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \qquad f(x) = 0$$

$$(1 \quad \alpha) \mapsto \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad f(x) = x$$

$$(\alpha \quad \alpha + 1) \mapsto \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \qquad f(x) = \alpha x$$

$$(\alpha + 1 \quad 1) \mapsto \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \qquad f(x) = (\alpha + 1)x$$

## Interpolation Decoding:

$\mathbf{c} = (f(g_1), \ldots, f(g_n)) \in \mathbb{F}_{q^m}^n$ codeword;
$\mathbf{r} = \mathbf{c} + \mathbf{e} \in \mathbb{F}_{q^m}^n$ received word.

**Theorem**

$d_R(\mathbf{c}, \mathbf{r}) = t$ if and only if there exists a $D(x) \in \mathcal{L}_q(x, q^m)$, such that $\operatorname{qdeg}(D(x)) = t$ and

$$D(f(g_i)) = D(r_i) \quad \forall i \in \{1, \ldots, n\}.$$

This $D(x)$ is called the *error span polynomial*.

## Interpolation Decoding:

$\mathbf{c} = (f(g_1), \ldots, f(g_n)) \in \mathbb{F}_{q^m}^n$ codeword;
$\mathbf{r} = \mathbf{c} + \mathbf{e} \in \mathbb{F}_{q^m}^n$ received word.

**Theorem**

$d_R(\mathbf{c}, \mathbf{r}) = t$ if and only if there exists a $D(x) \in \mathcal{L}_q(x, q^m)$, such that $\operatorname{qdeg}(D(x)) = t$ and

$$D(f(g_i)) = D(r_i) \quad \forall i \in \{1, \ldots, n\}.$$

This $D(x)$ is called the *error span polynomial*.

Set $N(x) := D(f(x))$. Then $N(g_i) - D(r_i) = 0$.

## Definition

Define $\mathfrak{M}(\mathbf{r})$ as the (left) module in $\mathcal{L}_q(x, q^m)^2$, that contains exactly all $[N(x) \quad -D(x)] \in \mathcal{L}_q(x, q^m)^2$ with $N(g_i) - D(r_i) = 0$. We call $\mathfrak{M}(\mathbf{r})$ the *interpolation module of* $\mathbf{r}$.

## Definition

Define $\mathfrak{M}(\mathbf{r})$ as the (left) module in $\mathcal{L}_q(x, q^m)^2$, that contains exactly all $[N(x) \quad -D(x)] \in \mathcal{L}_q(x, q^m)^2$ with $N(g_i) - D(r_i) = 0$. We call $\mathfrak{M}(\mathbf{r})$ the *interpolation module of* $\mathbf{r}$.

*Annihilator polynomial*:
$\Pi_{\mathbf{g}}(x) \in \mathcal{L}_q(x, q^m)$ such that $\Pi_{\mathbf{g}}(g_i) = 0$
*q-Lagrange polynomial*:
$\Lambda_{\mathbf{g},\mathbf{r}}(x) \in \mathcal{L}_q(x, q^m)$ such that $\Lambda_{\mathbf{g},\mathbf{r}}(g_i) = r_i$

## Theorem

$$\mathfrak{M}(\mathbf{r}) = \text{rowspan} \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}$$

**Theorem**

The elements $[N(x) \quad -D(x)]$ of $\mathfrak{M}(\mathbf{r})$ that fulfill

1. $\operatorname{qdeg}(N(x)) \leq t + k - 1$,
2. $\operatorname{qdeg}(D(x)) = t$,
3. $N(x)$ is symbolically divisible on the left by $D(x)$, i.e. there exists $f(x) \in \mathcal{L}_q(x, q^m)$ such that $D(f(x)) = N(x)$,

are in one-to-one correspondence with the codewords of rank distance $t$ to $\mathbf{r}$.

**Theorem**

The elements $[N(x) \quad -D(x)]$ of $\mathfrak{M}(\mathbf{r})$ that fulfill

1. $\operatorname{qdeg}(N(x)) \leq t + k - 1$,
2. $\operatorname{qdeg}(D(x)) = t$,
3. $N(x)$ is symbolically divisible on the left by $D(x)$, i.e. there exists $f(x) \in \mathcal{L}_q(x, q^m)$ such that $D(f(x)) = N(x)$,

are in one-to-one correspondence with the codewords of rank distance $t$ to $\mathbf{r}$.

The quotient is the respective message polynomial in $\mathcal{L}_q(x, q^m)$.

## The General Set-Up of the Algorithm:

Precomputed and stored: $\Pi_{\mathbf{g}}(x) \in \mathcal{L}_q(x, q^m)$ (annihilator)
Input: received word $\mathbf{r} \in \mathbb{F}_{q^m}^n$

## The General Set-Up of the Algorithm:

Precomputed and stored: $\Pi_{\mathbf{g}}(x) \in \mathcal{L}_q(x, q^m)$ (annihilator)
Input: received word $\mathbf{r} \in \mathbb{F}_{q^m}^n$

- Compute $q$-Lagrange polynomial $\Lambda_{\mathbf{g},\mathbf{r}}(x)$.
- Compute a minimal basis $\{b_1, b_2\}$ of the interpolation module

$$\mathfrak{M}(\mathbf{r}) = \text{rowspan} \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

- Check all elements of the form $\lambda(x) \circ b_1 + \mu(x) \circ b_2$ with restricted degrees of $\lambda(x), \mu(x) \in \mathcal{L}_q(x, q^m)$ for divisibility.

## The General Set-Up of the Algorithm:

Precomputed and stored: $\Pi_{\mathbf{g}}(x) \in \mathcal{L}_q(x, q^m)$ (annihilator)
Input: received word $\mathbf{r} \in \mathbb{F}_{q^m}^n$

- Compute $q$-Lagrange polynomial $\Lambda_{\mathbf{g},\mathbf{r}}(x)$.
- Compute a minimal basis $\{b_1, b_2\}$ of the interpolation module

$$\mathfrak{M}(\mathbf{r}) = \text{rowspan} \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

- Check all elements of the form $\lambda(x) \circ b_1 + \mu(x) \circ b_2$ with restricted degrees of $\lambda(x), \mu(x) \in \mathcal{L}_q(x, q^m)$ for divisibility.

Output: the symbolic quotients (where existent)

## The General Set-Up of the Algorithm:

Precomputed and stored: $\Pi_{\mathbf{g}}(x) \in \mathcal{L}_q(x, q^m)$ (annihilator)
Input: received word $\mathbf{r} \in \mathbb{F}_{q^m}^n$

- Compute $q$-Lagrange polynomial $\Lambda_{\mathbf{g},\mathbf{r}}(x)$.
- Compute a minimal basis $\{b_1, b_2\}$ of the interpolation module

$$\mathfrak{M}(\mathbf{r}) = \text{rowspan} \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

- Check all elements of the form $\lambda(x) \circ b_1 + \mu(x) \circ b_2$ with restricted degrees of $\lambda(x), \mu(x) \in \mathcal{L}_q(x, q^m)$ for divisibility.

Output: the symbolic quotients (where existent)

## The General Set-Up of the Algorithm:

Precomputed and stored: $\Pi_{\mathbf{g}}(x) \in \mathcal{L}_q(x, q^m)$ (annihilator)
Input: received word $\mathbf{r} \in \mathbb{F}_{q^m}^n$

- Compute $q$-Lagrange polynomial $\Lambda_{\mathbf{g},\mathbf{r}}(x)$.
- Compute a minimal basis $\{b_1, b_2\}$ of the interpolation module

$$\mathfrak{M}(\mathbf{r}) = \text{rowspan} \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

- Check all elements of the form $\lambda(x) \circ b_1 + \mu(x) \circ b_2$ with restricted degrees of $\lambda(x), \mu(x) \in \mathcal{L}_q(x, q^m)$ for divisibility.

Output: the symbolic quotients (where existent)

minimal basis of the interpolation module

How is minimality defined?

## Minimal Module Bases:

- $(0, k-1)$-weighted $q$-degree of $b_i = [N_i(x) \quad D_i(x)]$ is given by

$$\max\{ \operatorname{qdeg}(N_i(x)) \quad , \quad \operatorname{qdeg}(D_i(x)) + k - 1 \}$$

**Minimal Module Bases:**

- $(0, k-1)$-weighted $q$-degree of $b_i = [N_i(x) \quad D_i(x)]$ is given by

$$\max\{\ \mathrm{qdeg}(N_i(x)) \quad , \quad \mathrm{qdeg}(D_i(x)) + k - 1\ \}$$

- Module basis $\{b_1, b_2\}$ is minimal if

$$\mathrm{qdeg}(N_1(x)) \geq \mathrm{qdeg}(D_1(x)) + k - 1$$

$$\mathrm{qdeg}(N_2(x)) < \mathrm{qdeg}(D_2(x)) + k - 1$$

i.e. if the leading positions differ w.r.t. the $(0, k-1)$-weighted $q$-degree.

**Computation of minimal basis of $\mathfrak{M}(\mathbf{r})$ via EEA:**

Initialize $j = 0$ and

$$\begin{bmatrix} P_0(x) & K_0(x) \\ P_1(x) & K_1(x) \end{bmatrix} := \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

**while** $\operatorname{qdeg}(K_{j+1}) + k - 1 < \operatorname{qdeg}(P_{j+1})$ **do**

Apply symbolic division to get $q_j(x), r_j(x) \in \mathcal{L}_q(x, q^m)$ s.t. $P_j(x) = q_j(x) \circ P_{j+1}(x) + r_j(x)$ and $\operatorname{qdeg}(r_j) < \operatorname{qdeg}(P_{j+1})$.

$$\begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ P_{j+2}(x) & K_{j+2}(x) \end{bmatrix} := \begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ r_j(x) & K_j(x) - q_j(x) \circ K_{j+1}(x) \end{bmatrix}$$

Set $j := j + 1$.

**end while**

**return**

$b_1 := [\, P_{j+1}(x) \quad K_{j+1}(x) \,]$ and $b_2 := [\, P_{j+2}(x) \quad K_{j+2}(x) \,]$

**Computation of minimal basis of $\mathfrak{M}(\mathbf{r})$ via EEA:**

Initialize $j = 0$ and

$$\begin{bmatrix} P_0(x) & K_0(x) \\ P_1(x) & K_1(x) \end{bmatrix} := \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

**while** $\mathrm{qdeg}(K_{j+1}) + k - 1 < \mathrm{qdeg}(P_{j+1})$ **do**

Apply symbolic division to get $q_j(x), r_j(x) \in \mathcal{L}_q(x, q^m)$ s.t. $P_j(x) = q_j(x) \circ P_{j+1}(x) + r_j(x)$ and $\mathrm{qdeg}(r_j) < \mathrm{qdeg}(P_{j+1})$.

$$\begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ P_{j+2}(x) & K_{j+2}(x) \end{bmatrix} := \begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ r_j(x) & K_j(x) - q_j(x) \circ K_{j+1}(x) \end{bmatrix}$$

Set $j := j + 1$.

**end while**

**return**

$b_1 := [\, P_{j+1}(x) \quad K_{j+1}(x) \,]$ and $b_2 := [\, P_{j+2}(x) \quad K_{j+2}(x) \,]$

**Computation of minimal basis of $\mathfrak{M}(\mathbf{r})$ via EEA:**

Initialize $j = 0$ and

$$\begin{bmatrix} P_0(x) & K_0(x) \\ P_1(x) & K_1(x) \end{bmatrix} := \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

**while** $\mathrm{qdeg}(K_{j+1}) + k - 1 < \mathrm{qdeg}(P_{j+1})$ **do**

Apply symbolic division to get $q_j(x), r_j(x) \in \mathcal{L}_q(x, q^m)$ s.t. $P_j(x) = q_j(x) \circ P_{j+1}(x) + r_j(x)$ and $\mathrm{qdeg}(r_j) < \mathrm{qdeg}(P_{j+1})$.

$$\begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ P_{j+2}(x) & K_{j+2}(x) \end{bmatrix} := \begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ r_j(x) & K_j(x) - q_j(x) \circ K_{j+1}(x) \end{bmatrix}$$

Set $j := j + 1$.

**end while**

**return**

$b_1 := [\, P_{j+1}(x) \quad K_{j+1}(x) \,]$ and $b_2 := [\, P_{j+2}(x) \quad K_{j+2}(x) \,]$

**Computation of minimal basis of $\mathfrak{M}(\mathbf{r})$ via EEA:**

Initialize $j = 0$ and

$$\begin{bmatrix} P_0(x) & K_0(x) \\ P_1(x) & K_1(x) \end{bmatrix} := \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

**while** $\operatorname{qdeg}(K_{j+1}) + k - 1 < \operatorname{qdeg}(P_{j+1})$ **do**

Apply symbolic division to get $q_j(x), r_j(x) \in \mathcal{L}_q(x, q^m)$ s.t. $P_j(x) = q_j(x) \circ P_{j+1}(x) + r_j(x)$ and $\operatorname{qdeg}(r_j) < \operatorname{qdeg}(P_{j+1})$.

$$\begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ P_{j+2}(x) & K_{j+2}(x) \end{bmatrix} := \begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ r_j(x) & K_j(x) - q_j(x) \circ K_{j+1}(x) \end{bmatrix}$$

Set $j := j + 1$.

**end while**

**return**

$b_1 := [\, P_{j+1}(x) \quad K_{j+1}(x) \,]$ and $b_2 := [\, P_{j+2}(x) \quad K_{j+2}(x) \,]$

**Computation of minimal basis of $\mathfrak{M}(\mathbf{r})$ via EEA:**

Initialize $j = 0$ and

$$\begin{bmatrix} P_0(x) & K_0(x) \\ P_1(x) & K_1(x) \end{bmatrix} := \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

**while** $\operatorname{qdeg}(K_{j+1}) + k - 1 < \operatorname{qdeg}(P_{j+1})$ **do**

Apply symbolic division to get $q_j(x), r_j(x) \in \mathcal{L}_q(x, q^m)$ s.t. $P_j(x) = q_j(x) \circ P_{j+1}(x) + r_j(x)$ and $\operatorname{qdeg}(r_j) < \operatorname{qdeg}(P_{j+1})$.

$$\begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ P_{j+2}(x) & K_{j+2}(x) \end{bmatrix} := \begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ r_j(x) & K_j(x) - q_j(x) \circ K_{j+1}(x) \end{bmatrix}$$

Set $j := j + 1$.

**end while**

**return**

$b_1 := [\, P_{j+1}(x) \quad K_{j+1}(x) \,]$ and $b_2 := [\, P_{j+2}(x) \quad K_{j+2}(x) \,]$

Now what about the parametrization?

## The Parametrization:

Goal: Find all $[N(x) \quad -D(x)] \in \mathfrak{M}(\mathbf{r})$ with

1. $\operatorname{qdeg}(N(x)) \leq t + k - 1$,
2. $\operatorname{qdeg}(D(x)) = t$,

## The Parametrization:

Goal: Find all $[N(x) \quad -D(x)] \in \mathfrak{M}(\mathbf{r})$ with

1. $\mathrm{qdeg}(N(x)) \leq t + k - 1$,
2. $\mathrm{qdeg}(D(x)) = t$,

Solution: If $\{b_1, b_2\}$ is minimal basis (ordered by leading positions) with $\ell_1, \ell_2$ the respective weighted row $q$-degrees, then

$$\lambda(x) \circ b_1 + \mu(x) \circ b_2 \qquad \text{with}$$

1. $\mathrm{qdeg}(\lambda(x)) \leq t - \ell_1 + k - 1$,
2. $\mathrm{qdeg}(\mu(x)) = t - \ell_2 + k - 1$ and $\mu(x)$ is monic

yield all the desired elements.

## The Parametrization:

Goal: Find all $[N(x) \quad -D(x)] \in \mathfrak{M}(\mathbf{r})$ with

1. $\mathrm{qdeg}(N(x)) \leq t + k - 1$,
2. $\mathrm{qdeg}(D(x)) = t$,

Solution: If $\{b_1, b_2\}$ is minimal basis (ordered by leading positions) with $\ell_1, \ell_2$ the respective weighted row $q$-degrees, then

$$\lambda(x) \circ b_1 + \mu(x) \circ b_2 \qquad \text{with}$$

1. $\mathrm{qdeg}(\lambda(x)) \leq t - \ell_1 + k - 1$,
2. $\mathrm{qdeg}(\mu(x)) = t - \ell_2 + k - 1$ and $\mu(x)$ is monic

yield all the desired elements.

Proof: Based on the *Predictable Leading Monomial Property* for minimal bases of linearized polynomial modules.

**Example:**
Consider the Gabidulin code $C$ over $\mathbb{F}_{2^3} \cong \mathbb{F}_2[\alpha]$ (with $\alpha^3 = \alpha + 1$) with generator matrix

$$G = \begin{pmatrix} 1 & \alpha & \alpha^2 \\ 1 & \alpha^2 & \alpha^4 \end{pmatrix}.$$

$\implies d_R = n - k + 1 = 3 - 2 + 1 = 2$
$\implies C$ is no-error correcting

**Example:**

Consider the Gabidulin code $C$ over $\mathbb{F}_{2^3} \cong \mathbb{F}_2[\alpha]$ (with $\alpha^3 = \alpha + 1$) with generator matrix

$$G = \begin{pmatrix} 1 & \alpha & \alpha^2 \\ 1 & \alpha^2 & \alpha^4 \end{pmatrix}.$$

$\implies d_R = n - k + 1 = 3 - 2 + 1 = 2$

$\implies C$ is no-error correcting

Received word:

$$\mathbf{r} = (\, \alpha + 1 \quad 0 \quad \alpha \,).$$

**Example:**
Consider the Gabidulin code $C$ over $\mathbb{F}_{2^3} \cong \mathbb{F}_2[\alpha]$ (with $\alpha^3 = \alpha + 1$) with generator matrix

$$G = \begin{pmatrix} 1 & \alpha & \alpha^2 \\ 1 & \alpha^2 & \alpha^4 \end{pmatrix}.$$

$\implies d_R = n - k + 1 = 3 - 2 + 1 = 2$
$\implies C$ is no-error correcting
Received word:

$$\mathbf{r} = \begin{pmatrix} \alpha + 1 & 0 & \alpha \end{pmatrix}.$$

Interpolation module:

$$\mathfrak{M}(\mathbf{r}) = \operatorname{rowspan} \begin{bmatrix} \Pi_{\mathbf{g}}(x) & 0 \\ \Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix} = \operatorname{rowspan} \begin{bmatrix} x^8 + x & 0 \\ \alpha^2 x^4 + \alpha^5 x & x \end{bmatrix}.$$

Finding the minimal basis with the Euclidean algorithm:

$$x^8 + x = (\alpha^3 x^2) \circ (\alpha^2 x^4 + \alpha^5 x) + (\alpha^6 x^2 + x).$$

Since $\mathrm{qdeg}(\alpha^3 x^2) + k - 1 = 2 \geq 1 = \mathrm{qdeg}(\alpha^6 x^2 + x)$, the algorithm terminates and a minimal basis (w.r.t. the $(0,1)$-weighted 2-degree) of this module is

$$\begin{bmatrix} 0 & x \\ x & \alpha^3 x^2 \end{bmatrix} \circ \underbrace{\begin{bmatrix} x^8 + x & 0 \\ \alpha^2 x^4 + \alpha^5 x & x \end{bmatrix}}_{\text{original basis}} = \underbrace{\begin{bmatrix} \alpha^2 x^4 + \alpha^5 x & x \\ \alpha^6 x^2 + x & \alpha^3 x^2 \end{bmatrix}}_{\text{minimal basis}}.$$

Parametrization $\lambda(x) \circ b_1 + \mu(x) \circ b_2$:

Use all $\lambda(x) \in \mathcal{L}_2(x, 2^3)$ with 2-degree 0 and all monic $\mu(x) \in \mathcal{L}_2(x, 2^3)$ with 2-degree 0.

$$\implies \lambda(x) = a_0 x, \quad a_0 \in \mathbb{F}_{2^3}, \quad \text{and} \quad \mu(x) = x$$

Parametrization $\lambda(x) \circ b_1 + \mu(x) \circ b_2$:

Use all $\lambda(x) \in \mathcal{L}_2(x, 2^3)$ with 2-degree 0 and all monic $\mu(x) \in \mathcal{L}_2(x, 2^3)$ with 2-degree 0.

$$\implies \lambda(x) = a_0 x, \quad a_0 \in \mathbb{F}_{2^3}, \quad \text{and} \quad \mu(x) = x$$

E.g. for $a_0 = \alpha$:

$$(\alpha x) \circ [\alpha^2 x^4 + \alpha^5 x \qquad x] + [\alpha^6 x^2 + x \qquad \alpha^3 x^2]$$

$$= [\alpha^3 x^4 + \alpha^6 x \qquad \alpha x] + [\alpha^6 x^2 + x \qquad \alpha^3 x^2]$$

$$= [\alpha^3 x^4 + \alpha^6 x^2 + \alpha^2 x \qquad \alpha^3 x^2 + \alpha x]$$

Parametrization $\lambda(x) \circ b_1 + \mu(x) \circ b_2$:

Use all $\lambda(x) \in \mathcal{L}_2(x, 2^3)$ with 2-degree 0 and all monic $\mu(x) \in \mathcal{L}_2(x, 2^3)$ with 2-degree 0.

$$\implies \lambda(x) = a_0 x, \quad a_0 \in \mathbb{F}_{2^3}, \quad \text{and} \quad \mu(x) = x$$

E.g. for $a_0 = \alpha$:

$$(\alpha x) \circ [\alpha^2 x^4 + \alpha^5 x \qquad x] + [\alpha^6 x^2 + x \qquad \alpha^3 x^2]$$

$$= [\alpha^3 x^4 + \alpha^6 x \qquad \alpha x] + [\alpha^6 x^2 + x \qquad \alpha^3 x^2]$$

$$= [\alpha^3 x^4 + \alpha^6 x^2 + \alpha^2 x \qquad \alpha^3 x^2 + \alpha x]$$

Symbolic division:

$$\alpha^3 x^4 + \alpha^6 x^2 + \alpha^2 x = (\alpha^3 x^2 + \alpha x) \circ ( \quad \underbrace{x^2 + \alpha x}_{\text{message polynomial}} \quad )$$

We get divisibility for all $a_0 \in \mathbb{F}_{2^3} \backslash \{0\}$. Thus our list decoding algorithm finds the following list of message polynomials and corresponding codewords:

$$
\begin{array}{ll}
m_1(x) = x^2 + \alpha x & c_1 = (\,\alpha+1 \quad 0 \quad \alpha^2+1\,), \\
m_2(x) = \alpha^5 x^2 + \alpha^2 x & c_2 = (\,\alpha+1 \quad \alpha \quad \alpha\,), \\
m_3(x) = \alpha^3 x^2 + \alpha^4 x & c_3 = (\,\alpha^2+1 \quad 0 \quad \alpha^2\,), \\
m_4(x) = \alpha^4 x^2 & c_4 = (\,\alpha^2+\alpha \quad \alpha^2+1 \quad \alpha\,), \\
m_5(x) = \alpha^6 x^2 + \alpha^6 x & c_5 = (\,0 \quad \alpha+1 \quad 1\,), \\
m_6(x) = \alpha^2 x^2 + \alpha^3 x & c_6 = (\,\alpha^2+\alpha+1 \quad 0 \quad \alpha\,), \\
m_7(x) = \alpha x^2 + x & c_7 = (\,\alpha+1 \quad 1 \quad \alpha+1\,).
\end{array}
$$

All these codewords are rank distance 1 away from

$$\mathbf{r} = (\,\alpha+1\;0\;\alpha\,).$$

We get divisibility for all $a_0 \in \mathbb{F}_{2^3} \backslash \{0\}$. Thus our list decoding algorithm finds the following list of message polynomials and corresponding codewords:

$$
\begin{array}{ll}
m_1(x) = x^2 + \alpha x & c_1 = (\,\alpha+1 \quad 0 \quad \alpha^2+1), \\
m_2(x) = \alpha^5 x^2 + \alpha^2 x & c_2 = (\,\alpha+1 \quad \alpha \quad \alpha), \\
m_3(x) = \alpha^3 x^2 + \alpha^4 x & c_3 = (\,\alpha^2+1 \quad 0 \quad \alpha^2), \\
m_4(x) = \alpha^4 x^2 & c_4 = (\,\alpha^2+\alpha \quad \alpha^2+1 \quad \alpha), \\
m_5(x) = \alpha^6 x^2 + \alpha^6 x & c_5 = (\,0 \quad \alpha+1 \quad 1), \\
m_6(x) = \alpha^2 x^2 + \alpha^3 x & c_6 = (\,\alpha^2+\alpha+1 \quad 0 \quad \alpha), \\
m_7(x) = \alpha x^2 + x & c_7 = (\,\alpha+1 \quad 1 \quad \alpha+1).
\end{array}
$$

All these codewords are rank distance 1 away from

$$\mathbf{r} = (\,\alpha+1\;0\;\alpha\,).$$

Note: Hamming distance can be $1, 2$ or $3$.

## Complexity:

- Computing $\Lambda_{\mathbf{g},\mathbf{r}}(x)$: $\mathcal{O}_{q^m}(n^2)$
- Computing the minimal basis of $\mathfrak{M}(\mathbf{r})$ with Euclidean algorithm: $\mathcal{O}_{q^m}(n^3)$
- Computing the minimal basis of $\mathfrak{M}(\mathbf{r})$ iteratively (not in this talk): $\mathcal{O}_{q^m}(qn^2)$

## Complexity:

- Computing $\Lambda_{\mathbf{g},\mathbf{r}}(x)$: $\mathcal{O}_{q^m}(n^2)$
- Computing the minimal basis of $\mathfrak{M}(\mathbf{r})$ with Euclidean algorithm: $\mathcal{O}_{q^m}(n^3)$
- Computing the minimal basis of $\mathfrak{M}(\mathbf{r})$ iteratively (not in this talk): $\mathcal{O}_{q^m}(qn^2)$

- Parametrization: $\mathcal{O}_{q^m}((q^{m(2t+k-n)}+q)n^2)$
  - If $t \leq (n-k)/2$ (unique decoding), then polynomial.

  - If $t > (n-k)/2$, then exponential.

**Idea to improve parametrization (open problem):**

- When finding all $[N(x) \quad -D(x)] \in \mathfrak{M}(\mathbf{r})$ with the degree restrictions we can impose extra condition that the error span polynomial $D(x)$ has only distinct roots.

**Idea to improve parametrization (open problem):**

- When finding all $[N(x) \quad -D(x)] \in \mathfrak{M}(\mathbf{r})$ with the degree restrictions we can impose extra condition that the error span polynomial $D(x)$ has only distinct roots.
- Open problem: How to parametrize this condition?
- In Reed-Solomon case it can be done with a curve fitting algorithm (based on Wu's algorithm). This idea does not work in the linearized case.

## Summary and Conclusion

- Our algorithm is an algebraic decoding algorithm for general Gabidulin codes, that finds *all* codewords within the ball of radius $t$ around a given received word, for *any* decoding radius $t$.
- Can easily be altered to find *all* closest codewords to a given received word.

## Summary and Conclusion

- Our algorithm is an algebraic decoding algorithm for general Gabidulin codes, that finds *all* codewords within the ball of radius $t$ around a given received word, for *any* decoding radius $t$.
- Can easily be altered to find *all* closest codewords to a given received word.
- We used a module set-up and a parametrization based on minimal bases of the interpolation module.
- Complexity is exponential in code length iff the decoding radius is beyond the unique decoding radius.
- Nonetheless, it is still feasible for radii close to the unique decoding radius.

## Summary and Conclusion

- Our algorithm is an algebraic decoding algorithm for general Gabidulin codes, that finds *all* codewords within the ball of radius $t$ around a given received word, for *any* decoding radius $t$.
- Can easily be altered to find *all* closest codewords to a given received word.
- We used a module set-up and a parametrization based on minimal bases of the interpolation module.
- Complexity is exponential in code length iff the decoding radius is beyond the unique decoding radius.
- Nonetheless, it is still feasible for radii close to the unique decoding radius.

Thank you for your attention!