Noname manuscript No.
(will be inserted by the editor)

The Predictable Leading Monomial Property for Linearized Polynomials and Gabidulin List-Decoding

Margreta Kuijper and Anna-Lena Trautmann

the date of receipt and acceptance should be inserted later

Abstract We show how Gabidulin codes can be list decoded by using a parametrization approach. Our decoding algorithm computes a list of all closest codewords to a given received word. We consider a certain module, called the interpolation module, over the ring of linearized polynomials with respect to composition of polynomials. The Predictable Leading Monomial property for minimal bases of this interpolation module is stated and proved, which is then used as a key ingredient for our parametrization. The parametrization is based on a minimal basis for the interpolation module, which is why we furthermore formulate two subalgorithms, one using the extended Euclidean algorithm and an iterative one, for finding such a basis.

Keywords linearized polynomials · polynomial modules · minimal basis · Gabidulin codes · rank metric · list-decoding · parametrization

Mathematics Subject Classification (2000) 16D40 · 94B05 · 94B35

1 Introduction

Over the last decade there has been increased interest in Gabidulin codes, mainly because of their relevance to network coding [6,24]. Gabidulin codes are optimal rank-metric nonbinary codes over a field $\mathbb{F}_q^m$ (where q is a prime power). They were first derived by Gabidulin in [4] and independently by Delsarte in [3]. These codes can be seen as the q-analog of Reed-Solomon codes, using q-linearized polynomials instead of arbitrary polynomials. They are optimal in the sense that they are not only MDS codes with respect to the Hamming metric, but also achieve the Singleton bound with respect to the rank metric and are thus MRD codes. They are not only of interest in network coding but also in space-time coding [14], crisscoss error correction [18] and distributed storage [21].

Department of Electrical and Electronic Engineering, University of Melbourne, Australia. ALT is also with the Department of Electrical and Computer Systems Engineering, Monash University. She was supported by Swiss National Science Foundation Fellowship no. 147304. E-mail: anna-lena.trautmann@unimelb.edu.au

The decoding of Gabidulin codes has obtained a fair amount of attention in the literature, starting with work on decoding inside the unique decoding radius in [4,5] and more recently [13,17,19,20,22]. Decoding beyond the unique decoding radius was investigated in e.g. [6,12,15,27,28]. Related work on list-decoding of lifted Gabidulin codes can be found in [25].

Using the close resemblance between Reed-Solomon codes and Gabidulin codes, the paper [13] translates Gabidulin decoding into a set of polynomial interpolation conditions. Essentially, this setup is also used in the papers [6,28] that present iterative algorithms that perform Gabidulin list decoding with a list size of 1. In this paper we present an iterative algorithm that bears similarity to the ones in [6, 13,28] but yields *all* closest codewords rather than just one. The latter is due to our parametrization approach.

The paper is structured as follows. In the next section we present several preliminaries on q-linearized polynomials, Gabidulin codes, the rank metric and we recall the polynomial interpolation conditions from [13]. We also detail an iterative construction of the q-annihilator polynomial and the q-Lagrange polynomial. Section 3 deals with modules over the ring of linearized polynomials and gives the Predictable Leading Monomial property for special types of bases of these modules. In Section 4 we reformulate the Gabidulin list decoding requirements in terms of a module represented by four q-linearized polynomials and present the decoding algorithm, which is based on a parametrization using the Predictable Leading Monomial property. For this we present two subalgorithms for computing a minimal basis of the interpolation module. Furthermore, we analyze the complexity of our algorithms. We conclude this paper in Section 5.

2 Preliminaries

2.1 q-linearized polynomials

Let q be a prime power and let $\mathbb{F}_q$ denote the finite field with q elements. It is well-known that there always exists a primitive element α of the extension field $\mathbb{F}_{q^m}$, such that $\mathbb{F}_{q^m} \cong \mathbb{F}_q[\alpha]$. Moreover, $\mathbb{F}_{q^m}$ is isomorphic (as a vector space) to the vector space $\mathbb{F}_q^m$. One then easily gets the isomorphic description of matrices over the base field $\mathbb{F}_q$ as vectors over the extension field, i.e. $\mathbb{F}_q^{m\times n} \cong \mathbb{F}_{q^m}^n$. Since we will work with matrices over different underlying fields we denote the rank of a matrix X over $\mathbb{F}_q$ by $\mathrm{rank}_q(X)$.

For some vector $(v_1,\dots,v_n) \in \mathbb{F}_{q^m}^n$ we denote the $k \times n$ *Moore matrix* by

$$M_k(v_1,\dots,v_n) := \begin{pmatrix} v_1 & v_2 & \dots & v_n \\ v_1^{[1]} & v_2^{[1]} & \dots & v_n^{[1]} \\ \vdots & & & \\ v_1^{[k-1]} & v_2^{[k-1]} & \dots & v_n^{[k-1]} \end{pmatrix},$$

where $[i] := q^i$. A *q-linearized polynomial* over $\mathbb{F}_{q^m}$ is defined to be of the form

$$f(x) = \sum_{i=0}^{n} a_i x^{[i]} \quad , \quad a_i \in \mathbb{F}_{q^m},$$

where n is called the *q-degree* of $f(x)$, assuming that $a_n \neq 0$, denoted by $\mathrm{qdeg}(f)$. This class of polynomials was first studied by Ore in [16]. One can easily check that $f(x_1 + x_2) = f(x_1) + f(x_2)$ and $f(\lambda x_1) = \lambda f(x_1)$ for any $x_1, x_2 \in \mathbb{F}_{q^m}$ and $\lambda \in \mathbb{F}_q$, hence the name *linearized*. The set of all q-linearized polynomials over $\mathbb{F}_{q^m}$ is denoted by $\mathcal{L}_q(x, q^m)$. This set forms a non-commutative ring with the normal addition $+$ and composition $\circ$ of polynomials. Because of the non-commutativity, products and quotients of elements of $\mathcal{L}_q(x, q^m)$ have to be specified as being "left" or 'right" products or quotients. To not be mistaken with the standard division, we call the inverse of the composition *symbolic division*. I.e. $f(x)$ is symbolically divisible by $g(x)$ with right quotient $m(x)$ if

$$g(x) \circ m(x) = g(m(x)) = f(x).$$

Efficient algorithms for all these operations (left and right symbolic multiplication and division) exist and can be found e.g. in [6].

Lemma 1 (cf. [11] Thm. 3.50) *Let $f(x) \in \mathcal{L}_q(x, q^m)$ and $\mathbb{F}_{q^s}$ be the smallest extension field of $\mathbb{F}_{q^m}$ that contains all roots of $f(x)$. Then the set of all roots of $f(x)$ forms a $\mathbb{F}_q$-linear vector space in $\mathbb{F}_{q^s}$.*

Lemma 2 ([11] Thm. 3.52) *Let U be a $\mathbb{F}_q$-linear subspace of $\mathbb{F}_{q^m}$. Then $\prod_{g \in U}(x - g)$ is an element of $\mathcal{L}_q(x, q^m)$.*

Note that, if $g_1, \ldots, g_n$ is a basis of U, one can rewrite

$$\prod_{g \in U}(x - g) = \lambda \det(M_{n+1}(g_1, \ldots, g_n, x))$$

for some constant $\lambda \in \mathbb{F}_{q^m}$. We call this polynomial the *q-annihilator polynomial of* U, denoted by $\Pi_{(g_1, g_2, \ldots, g_n)}(x)$. Clearly its q-degree equals n.

We also have a notion of q-Lagrange polynomial: Let $\mathbf{g} = (g_1, \ldots, g_n)$ and $\mathbf{r} = (r_1, \ldots, r_n)$, both in $\mathbb{F}_{q^m}^n$. Define the matrix $\mathfrak{D}_i(\mathbf{g}, x)$ as $M_n(g_1, \ldots, g_n, x)$ without the i-th column. We define the *q-Lagrange polynomial* as

$$\Lambda_{\mathbf{g},\mathbf{r}}(x) := \sum_{i=1}^{n} (-1)^{n-i} r_i \frac{\det(\mathfrak{D}_i(\mathbf{g}, x))}{\det(M_n(\mathbf{g}))} \quad \in \mathbb{F}_{q^m}[x].$$

It can be easily verified that the above polynomial is q-linearized and that $\Lambda_{\mathbf{g},\mathbf{r}}(g_i) = r_i$ for $i = 1, \ldots, n$.

Note that, although not under the same name, the previous two polynomials were also defined in e.g. [26].

In the following we will use matrix composition, which is defined analogously to matrix multiplication:

$$\begin{bmatrix} a(x) & b(x) \\ c(x) & d(x) \end{bmatrix} \circ \begin{bmatrix} e(x) & f(x) \\ g(x) & h(x) \end{bmatrix} :=$$

$$\begin{bmatrix} a(e(x)) + b(g(x)) & a(f(x)) + b(h(x)) \\ c(e(x)) + d(g(x)) & c(f(x)) + d(h(x)) \end{bmatrix}.$$

We can recursively construct the q-annihilator and the q-Lagrange polynomial as follows.

Proposition 3 *Let $g_1, \ldots, g_n \in \mathbb{F}_{q^m}$ be linearly independent and $r_1, \ldots, r_n \in \mathbb{F}_{q^m}$. Define*

$$\Pi_1(x) := x^q - g_1^{q-1}x \quad , \quad \Lambda_1(x) := \frac{r_1}{g_1}x,$$

and for $i = 1, \ldots, n-1$

$$\begin{bmatrix} \Pi_{i+1}(x) \\ \Lambda_{i+1}(x) \end{bmatrix} := \begin{bmatrix} x^q - \Pi_i(g_{i+1})^{q-1}x & 0 \\ -\frac{\Lambda_i(g_{i+1})-r_{i+1}}{\Pi_i(g_{i+1})}x & x \end{bmatrix} \circ \begin{bmatrix} \Pi_i(x) \\ \Lambda_i(x) \end{bmatrix}.$$

Then we have $\Pi_i(x) = \Pi_{(g_1,g_2,\ldots,g_i)}(x)$ and $\Lambda_i(x) = \Lambda_{(g_1,g_2,\ldots,g_i),(r_1,\ldots,r_i)}(x)$ for $i = 1, \ldots, n$.

Proof We prove this by induction on i. The theorem clearly holds for $i = 1$. Suppose that the theorem holds for a value of i with $1 \leq i < n$. By definition $\Pi_{i+1}(x) = \Pi_i(x)^q - \Pi_i(g_{i+1})^{q-1}\Pi_i(x)$, so that (using the induction hypothesis) $\Pi_{i+1}(x)$ is a monic q-linearized polynomial of q-degree $i+1$ such that for $1 \leq j \leq i+1$ we have $\Pi_{i+1}(g_j) = 0$. It follows that then $\Pi_{i+1}(x)$ must coincide with $\Pi_{(g_1,g_2,\ldots,g_{i+1})}(x)$.

We next show that the formula for $\Lambda_{i+1}(x)$ yields the q-Lagrange polynomial at level $i+1$. Assume that $\Lambda_i(x)$ is the q-Lagrange polynomial at level i and look at $\Lambda_{i+1}(x)$, which is q-linearized since $\Lambda_i(x)$ and $\Pi_i(x)$ are q-linearized. As $\operatorname{qdeg}(\Pi_i(x)) = i > \operatorname{qdeg}(\Lambda_i(x))$ it holds that $\operatorname{qdeg}(\Lambda_{i+1}(x)) = i$. Furthermore, because $\Pi_i(g_j) = 0$ for $j = 1, \ldots, i$ and $\Lambda_i(g_j) = r_j$ for $j = 1, \ldots, i$,

$$\Lambda_{i+1}(g_j) = \Lambda_i(g_j) = r_j, \quad \text{and}$$

$$\Lambda_{i+1}(g_{i+1}) = \Lambda_i(g_{i+1}) - \frac{\Lambda_i(g_{i+1}) - r_{i+1}}{\Pi_i(g_{i+1})}\Pi_i(g_{i+1}) = r_{i+1}.$$

Therefore, $\Lambda_{i+1}(x)$ evaluates to the same values as $\Lambda_{(g_1,\ldots,g_{i+1}),(r_1,\ldots,r_{i+1})}(x)$ for $g_1, \ldots, g_{i+1}$. Because of the linearity of both these polynomials they evaluate to the same values for all elements of $\langle g_1, \ldots, g_{i+1}\rangle$, and as the g_i are linearly independent, these are q^{i+1} many values. Since the degree of both polynomials is $q^i < q^{i+1}$, it follows that they must be the same polynomial.

Furthermore we need the following fact for our investigations in Section 4.

Lemma 4 ([10]) *Let $L(x) \in \mathcal{L}_q(x, q^m)$, such that $L(g_i) = 0$ for all i. Then*

$$\exists H(x) \in \mathcal{L}_q(x, q^m) : L(x) = H(x) \circ \prod_{g \in \langle g_1, \ldots, g_n\rangle} (x - g).$$

Proof We know from Lemma 2 that $\prod_{\alpha \in \langle g_1,\ldots,g_n\rangle}(x - \alpha) \in \mathcal{L}_q(x, q^m)$. Moreover there always exists unique left and right division in $\mathcal{L}_q(x, q^m)$, i.e. in this case there exist unique polynomials $H(x), R(x) \in \mathcal{L}_q(x, q^m)$ such that $L(x) = H(x) \circ \prod_{\alpha \in \langle g_1,\ldots,g_n\rangle}(x-\alpha) + R(x)$ and $\operatorname{qdeg}(R(x)) < \operatorname{qdeg}\prod_{\alpha \in \langle g_1,\ldots,g_n\rangle}(x-\alpha) = n$. Since any $\alpha \in \langle g_1, \ldots, g_n\rangle$ is a root of $L(x)$ and of $\prod_{\alpha \in \langle g_1,\ldots,g_n\rangle}(x-\alpha)$, they must also be a root of $R(x)$. Hence we have q^n distinct roots for $R(x)$ and $\deg(R) < q^n$, thus $R(x) \equiv 0$ and the statement follows.

2.2 Gabidulin codes

Let $g_1, \ldots, g_n \in \mathbb{F}_{q^m}$ be linearly independent over $\mathbb{F}_q$. We define a *Gabidulin code* $C \subseteq \mathbb{F}_{q^m}^n$ as the linear block code with generator matrix $M_k(g_1, \ldots, g_n)$. Using the isomorphic matrix representation we can interpret C as a matrix code in $\mathbb{F}_q^{m \times n}$.The *rank distance* d_R on $\mathbb{F}_q^{m \times n}$ is defined by

$$d_R(X, Y) := \mathrm{rank}_q(X - Y) \quad , \quad X, Y \in \mathbb{F}_q^{m \times n}$$

and analogously for the isomorphic extension field representation. It holds that the code C constructed before has dimension k over $\mathbb{F}_{q^m}$ and minimum rank distance (over $\mathbb{F}_q$) $n - k + 1$. One can easily see by the shape of the parity check and the generator matrices that an equivalent definition of the code is

$$C = \{(m(g_1), \ldots, m(g_n)) \in \mathbb{F}_{q^m}^n \mid m(x) \in \mathcal{L}_q(x, q^m)_{<k}\},$$

where $\mathcal{L}_q(x, q^m)_{<k} := \{m(x) \in \mathcal{L}_q(x, q^m), \mathrm{qdeg}(m(x)) < k\}$. For more information on bounds and constructions of rank-metric codes the interested reader is referred to [4].

Consider a received word $\mathbf{r} = (r_1, \ldots, r_n) \in \mathbb{F}_{q^m}^n$ as the sum $\mathbf{r} = \mathbf{c} + \mathbf{e}$, where $\mathbf{c} = (c_1, \ldots, c_n) \in C$ is a codeword and $\mathbf{e} = (e_1, \ldots, e_n) \in \mathbb{F}_{q^m}^n$ is the error vector. We now recall the polynomial interpolation setup from [13] via a more general formulation in the next theorem.

Theorem 5 ([10,13]) *Let $m(x) \in \mathcal{L}_q(x, q^m), \mathrm{qdeg}(f(x)) < k$ and $c_i = m(g_i)$ for $i = 1, \ldots, n$. Then $d_R(\mathbf{c}, \mathbf{r}) = t$ if and only if there exists a $D(x) \in \mathcal{L}_q(x, q^m)$, such that $\mathrm{qdeg}(D(x)) = t$ and*

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

Furthermore, this $D(x)$ is unique.

Remark 6 *The previous theorem states that the roots of $D(x)$ form a vector space of degree t which is equal to the span of $e_1, \ldots, e_n$ (for this note that $e_i = m(g_i) - r_i$). This is why $D(x)$ is also called the* error span polynomial *(cf. e.g. [23]). The analogy in the classical Hamming metric set-up is the* error locator polynomial*, whose roots indicate the locations of the errors, and whose degree equals the number of errors.*

3 Modules over $\mathcal{L}_q(x, q^m)$

As mentioned before, $\mathcal{L}_q(x, q^m)$ forms a ring with addition and composition. Hence $\mathcal{L}_q(x, q^m)^\ell$ forms a (right or left) module. In this work we will consider $\mathcal{L}_q(x, q^m)^\ell$ as a left module and investigate its (left) submodules.

We will first give some general definitions and results on $\mathcal{L}_q(x, q^m)^\ell$ and then state and prove the Predictable Leading Monomial property in the end of this section. All of these are analogous to the definitions and results for modules over $\mathbb{F}_{q^m}[x]$ (equipped with normal polynomial multiplication) from [2]. To avoid confusion we denote polynomials by $f(x)$, while vectors of polynomials are denoted by f. If we need to index polynomials, we use the notation $f_1(x), \ldots, f_s(x)$, while for vectors of polynomials we will use the notation $f^{(1)}, \ldots, f^{(s)}$.

Elements of $\mathcal{L}_q(x,q^m)^\ell$ are of the form

$$f := [f_1(x) \ \dots \ f_\ell(x)] = \sum_{i=1}^{\ell} f_i(x)e_i$$

where $f_i(x) = \sum_j f_{ij}x^{[j]} \in \mathcal{L}_q(x,q^m)$ and $e_1,\dots,e_\ell$ are the unit vectors of length ℓ. Analogous to polynomial multiplication on $\mathbb{F}_{q^m}[x]^\ell$ we define for $h(x) \in \mathcal{L}_q(x,q^m)$ the left operation

$$h(x) \circ f := [h(f_1(x)) \ \dots \ h(f_\ell(x))] = \sum_{i=1}^{\ell} h(f_i(x))e_i.$$

The monomials of f are of the form $x^{[k]}e_i$ for all k such that $f_{ik} \neq 0$.

Definition 7 *A subset $M \subseteq \mathcal{L}_q(x,q^m)^\ell$ is a* (left) submodule *of $\mathcal{L}_q(x,q^m)^\ell$ if it is closed under addition and composition with $\mathcal{L}_q(x,q^m)$ on the left.*

Definition 8 *Consider the non-zero elements $f^{(1)},\dots,f^{(s)} \in \mathcal{L}_q(x,q^m)^\ell$. We say that $f^{(1)},\dots,f^{(s)}$ are* linearly independent *if for any $a_1(x),\dots,a_s(x) \in \mathcal{L}_q(x,q^m)$*

$$\sum_{i=1}^{s} a_i(x) \circ f^{(i)} = [\,0 \ \dots \ 0\,] \quad \Longrightarrow \quad a_1(x) = \cdots = a_s(x) = 0.$$

A generating set of a submodule $M \subseteq \mathcal{L}_q(x,q^m)^\ell$ is called a basis *of M if all its elements are linearly independent.*

One can easily see that

$$B = \{xe_1, xe_2 \dots, xe_\ell\}$$

is a basis of $\mathcal{L}_q(x,q^m)^\ell$, thus $\mathcal{L}_q(x,q^m)^\ell$ is a *free* and *finitely generated* module.

We need the notion of monomial order for the subsequent results, which we will define in analogy to [1, Definition 3.5.1].

Definition 9 *A* monomial order $<$ *on $\mathcal{L}_q(x,q^m)^\ell$ is a total order on $\mathcal{L}_q(x,q^m)^\ell$ that fulfills the following two conditions:*

- *$x^{[k]}e_i < x^{[j]} \circ (x^{[k]}e_i)$ for any monomial $x^{[k]}e_i \in \mathcal{L}_q(x,q^m)^\ell$ and $j \in \mathbb{N}_{>0}$.*
- *If $x^{[k]}e_i < x^{[k']}e_{i'}$, then $x^{[j]} \circ (x^{[k]}e_i) < x^{[j]} \circ (x^{[k']}e_{i'})$ for any monomials $x^{[k]}e_i, x^{[k']}e_{i'} \in \mathcal{L}_q(x,q^m)^\ell$ and $j \in \mathbb{N}_0$.*

We have different choices for monomial orders, of which the following is of interest for our investigations.

Definition 10 *The* $(k_1,\dots,k_\ell)$-weighted term-over-position monomial order *is defined as*

$$x^{[i_1]}e_{j_1} <_{(k_1,\dots,k_2)} x^{[i_2]}e_{j_2} :\Longleftrightarrow$$

$$i_1 + k_{j_1} < i_2 + k_{j_2} \text{ or } [i_1 + k_{j_1} = i_2 + k_{j_2} \text{ and } j_1 < j_2].$$

Note that this monomial order for $\mathcal{L}_q(x, q^m)^\ell$ coincides with the weighted term-over-position monomial order for $\mathbb{F}_{q^m}[x]$, since one could replace the q-degrees with normal degrees and get the classical cases.

We furthermore need the following definition in analogy to the weighted term-over-position monomial order:

Definition 11 *The* $(k_1, \ldots, k_\ell)$-weighted q-degree *of* $[f_1(x) \ \ldots \ f_\ell(x)]$ *is defined as* $\max\{k_i + \mathrm{qdeg}(f_i(x)) \mid i = 1, \ldots, \ell\}$.

In the following we will not fix a monomial order. The results, if not noted differently, hold for any chosen monomial order.

Definition 12 *We can order all monomials of an element* $f \in \mathcal{L}_q(x, q^m)^\ell$ *in decreasing order with respect to some monomial order. Rename them such that* $x^{[i_1]}e_{j_1} > x^{[i_2]}e_{j_2} > \ldots$. *Then*

1. *the* leading monomial $\mathrm{lm}(f) = x^{[i_1]}e_{j_1}$ *is the greatest monomial of* f.
2. *the* leading position $\mathrm{lpos}(f) = j_1$ *is the vector coordinate of the leading monomial.*
3. *the* leading term $\mathrm{lt}(f) = f_{j_1,i_1}x^{[i_1]}e_{j_1}$ *is the complete term of the leading monomial.*

In order to define minimality for submodule bases we need the following notion of reduction, in analogy to [1, Definition 4.1.1].

Definition 13 *Let* $f, h \in \mathcal{L}_q(x, q^m)^\ell$ *and let* $F = \{f^{(1)}, \ldots, f^{(s)}\}$ *be a set of non-zero elements of* $\mathcal{L}_q(x, q^m)^\ell$. *We say that* f reduces to h modulo F in one step *if and only if*

$$h = f - ((b_1 x^{[a_1]}) \circ f^{(1)} + \cdots + (b_k x^{[a_k]}) \circ f^{(k)})$$

for some $a_1, \ldots, a_k \in \mathbb{N}_0$ *and* $b_1, \ldots, b_k \in \mathbb{F}_{q^m}$, *where*

$$\mathrm{lm}(f) = x^{[a_i]} \circ \mathrm{lm}(f^{(i)}), \quad i = 1, \ldots, k, \quad \text{and}$$

$$\mathrm{lt}(f) = (b_1 x^{[a_1]}) \circ \mathrm{lt}(f^{(1)}) + \cdots + (b_k x^{a_{[k]}}) \circ \mathrm{lt}(f^{(k)}).$$

We say that f *is* minimal *with respect to* F *if it cannot be reduced modulo* F.

Definition 14 *A module basis* B *is called* minimal *if all its elements* b *are minimal with respect to* $B \backslash \{b\}$.

Proposition 15 *Let* B *be a basis of a module* $M \subseteq \mathcal{L}_q(x, q^m)^\ell$. *Then* B *is a minimal basis if and only if all leading positions of the elements of* B *are distinct.*

Proof Let B be minimal. If two elements of B have the same leading position, the one with the greater leading monomial can be reduced modulo the other element, which contradicts the minimality. Hence, no two elements of a minimal basis can have the same leading position.

The other direction follows straight from the definition of reducibility and minimality of a basis, since if the leading positions of all elements are different, none of them can be reduced modulo the other elements.

The property outlined in the following theorem is called the *Predictable Leading Monomial (PLM) property*, a terminology that was introduced in [7] for modules in $\mathbb{F}_q[x]^\ell$ with respect to multiplication. Note that in [7] minimal bases were addressed as minimal Gröbner bases. It can be shown that in their setting as well as in the one of this paper a minimal basis is the same as a minimal Gröbner basis. For the theory of Gröbner bases for modules in $\mathcal{L}_q(x,q^m)^\ell$ the interested reader is referred to [8].

Theorem 16 (PLM property) *Let M be a module in $\mathcal{L}_q(x,q^m)^\ell$ with minimal basis $B=\{b^{(1)},\dots,b^{(k)}\}$. Then for any $0\neq f\in M$, written as*

$$f=a_1(x)\circ b^{(1)}+\cdots+a_k(x)\circ b^{(k)},$$

where $a_1(x),\dots,a_k(x)\in\mathcal{L}_q(x,q^m)$, we have

$$\operatorname{lm}(f)=\max_{1\leq i\leq k;a_i(x)\neq 0}\{\operatorname{lm}(a_i)\circ\operatorname{lm}(b^{(i)})\}$$

where (with slight abuse of notation) $\operatorname{lm}(a_i(x))$ denotes the term of $a_i(x)$ of highest q-degree.

Proof Since B is minimal, all leading positions and thus also all leading monomials of its elements are distinct (by Proposition 15). Without loss of generality assume that $\operatorname{lm}(b^{(1)})>\operatorname{lm}(b^{(2)})>\cdots>\operatorname{lm}(b^{(k)})$ and that all $a_i(x)$ are non-zero. Since $\mathcal{L}_q(x,q^m)$ contains no zero divisors, we have that $\operatorname{lpos}(a_i(x)\circ b^{(i)})=\operatorname{lpos}(b^{(i)})$ for $1\leq i\leq k$. As a result, all leading positions and therefore all leading monomials of the $a_i(x)\circ b^{(i)}$'s are distinct. Thus there exist $j_1,\dots,j_k$ such that

$$\operatorname{lm}(a_{j_1}(x)\circ b^{(j_1)})>\operatorname{lm}(a_{j_2}(x)\circ b^{(j_2)})>\cdots>\operatorname{lm}(a_{j_k}(x)\circ b^{(j_k)}).$$

It follows that

$$\operatorname{lm}(f)=\operatorname{lm}(a_{j_1}(x)\circ b^{(j_1)})=\operatorname{lm}(a_{j_1}(x))\circ\operatorname{lm}(b^{(j_1)})=\max_{1\leq i\leq k}\{\operatorname{lm}(a_i(x))\circ\operatorname{lm}(b^{(j_i)})\}.$$

Proposition 17 *The leading positions and weighted q-degrees of all elements of two distinct minimal bases for the same module in $\mathcal{L}_q(x,q^m)$ have to be the same. This implies that the cardinality of both bases are equal as well.*

Proof Let $B_1=\{b^{(i)}\mid i=1,\dots,k\}$ and $B_2=\{c^{(i)}\mid i=1,\dots,k'\}$ be two different minimal bases of the same module in $\mathcal{L}_q(x,q^m)$. Then $b^{(j)}$ must be a linear combination of the $c^{(i)}$ for $j=1,\dots,k$. Similarly, $c^{(i)}$ must be a linear combination of the $b^{(j)}$ for $i=1,\dots,k'$. Hence, by the PLM property and since all leading positions are different in the bases, there exist $j'\in\{1,\dots,k'\}$ and $a(x),a'(x)\in\mathcal{L}_q(x,q^m)$ such that $\operatorname{lm}(a(x)\circ c^{(j')})=\operatorname{lm}(b^{(j)})$ and $\operatorname{lm}(a'(x)\circ b^{(j)})=\operatorname{lm}(c^{(j')})$. This implies on the one hand that $\operatorname{lpos}(b^{(j)})=\operatorname{lpos}(c^{(j')})$ and on the other that $\operatorname{qdeg}(a(x))=\operatorname{qdeg}(a'(x))=0$, which implies that $\operatorname{qdeg}(b^{(j)})=\operatorname{qdeg}(c^{(j')})$.

4 Minimal List-Decoding of Gabidulin Codes

In this section we will describe a minimal list-decoding algorithm for Gabidulin codes. First we will derive a general algorithm using a parametrization approach inside the interpolation module for the given received word. To do so we will need a minimal basis with respect to the $(0, k-1)$-weighted q-degree of the interpolation module. How this minimal basis can be computed will be described in the second subsection.

For the remainder of the paper let $g_1, \ldots, g_n \in \mathbb{F}_{q^m}$ be linearly independent over $\mathbb{F}_q$ and let $M_k(g_1, \ldots, g_n)$ be the generator matrix of the Gabidulin code $C \subseteq \mathbb{F}_{q^m}^n$. Let $\mathbf{r} = (r_1, \ldots, r_n) \in \mathbb{F}_{q^m}^n$ be the received word and denote $\mathbf{g} = (g_1, \ldots, g_n)$. We will always use the $(0, k-1)$-weighted term-over-position monomial order as monomial order.

4.1 The Parametrization

Most of the results in this subsection can be found in our previous paper [10], although in that work they were not formulated in terms of minimal module bases. Moreover, due to space limitations some proofs could not completely be conducted in that work, therefore we will now write all proofs in detail and with respect to the results derived in Section 3.

In the following we abbreviate the row span of a (polynomial) matrix A by $\mathrm{rs}(A)$.

Definition 18 *The* interpolation module $\mathfrak{M}(\mathbf{r})$ *for* $\mathbf{r}$ *is defined as the left submodule of* $\mathcal{L}_q(x, q^m)$*, given by*

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

We identify any $[f(x) \quad g(x)] \in \mathfrak{M}(\mathbf{r})$ with the bivariate linearized q-polynomial $Q(x, y) = f(x) + g(y)$. The following theorem shows that the name interpolation module is justified for $\mathfrak{M}(\mathbf{r})$:

Theorem 19 $\mathfrak{M}(\mathbf{r})$ *consists exactly of all* $Q(x, y) = f(x) + g(y)$ *with* $f(x), g(x) \in \mathcal{L}_q(x, q^m)$*, such that* $Q(g_i, r_i) = 0$ *for* $i = 1, \ldots, n$*.*

Proof For the first direction let $Q(x, y) = f(x) + g(y)$ be an element of $\mathfrak{M}(\mathbf{r})$. Then there exist $\beta(x), \gamma(x) \in \mathcal{L}_q(x, q^m)$ such that $f(x) = \beta(x) \circ \Pi(x) - \gamma(x) \circ \Lambda_{\mathbf{g},\mathbf{r}}(x)$ and $\gamma(x) = g(x)$, thus $Q(g_i, r_i) = \beta(\Pi(g_i)) - b(\Lambda_{\mathbf{g},\mathbf{r}}(g_i)) + \gamma(r_i) = 0 - \gamma(r_i) + \gamma(r_i) = 0$.

For the other direction let $f(x), g(x) \in \mathcal{L}_q(x, q^m)$ be such that $Q(g_i, r_i) = f(g_i) + g(r_i) = 0$ for $i = 1, \ldots, n$. To show that $Q(x, y) \in \mathfrak{M}(\mathbf{r})$ we need to find $\beta(x) \in \mathcal{L}_q(x, q^m)$ such that

$$\beta(x) \circ \Pi(x) - \gamma(x) \circ \Lambda_{\mathbf{g},\mathbf{r}}(x) = f(x) \quad \text{and} \quad \gamma(x) = g(x).$$

We substitute the second into the first equation to get

$$\beta(x) \circ \Pi(x) = f(x) + g(x) \circ \Lambda_{\mathbf{g},\mathbf{r}}(x). \tag{1}$$

By assumption it holds that $f(g_i) + g(\Lambda_{\mathbf{g},\mathbf{r}}(g_i)) = f(g_i) + g(r_i) = 0$ for all i. Then, by Lemma 4, it follows that $f(x) + g(x) \circ \Lambda_{\mathbf{g},\mathbf{r}}(x)$ is symbolically divisible on the right by $\Pi(x)$ and hence there exists $\beta(x) \in \mathcal{L}_q(x, q^m)$ such that (1) holds.

Combining all the previous results we get a description of all codewords with distance t to the received word in the new parametrization:

Theorem 20 *The elements* $f = [N(x) \quad -D(x)]$ *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* $m(x) \in \mathcal{L}_q(x, q^m)$ *such that* $D(m(x)) = N(x)$,

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

Note that conditions 1) and 2) are equivalent to that the $(0, k-1)$*-weighted* q*-degree of* f *is equal to* $t + k - 1$ *and* $\operatorname{lpos}(f) = 2$.

Proof To prove the first direction let $\mathbf{c} \in \mathbb{F}_{q^m}^n$ be a codeword such that $d_R(\mathbf{c}, \mathbf{r}) = t$ with the corresponding message polynomial $m(x) \in \mathcal{L}_q(x, q^m)_{<k}$. Then by Theorem 5 there exists $D(x) \in \mathcal{L}_q(x, q^m)$ of q-degree t such that $D(m(g_i)) = D(r_i)$ for $i = 1, \ldots, n$. By Theorem 19 we know that $[D(m(x)) \quad -D(x)]$ is in $\mathfrak{M}(\mathbf{r})$. It holds that $\operatorname{qdeg}(D(m(x))) \leq t + k - 1$ and that $(D(m(x))$ is symbolically divisible on the left by $D(x)$.

For the other direction let $[N(x) \quad -D(x)] \in \mathfrak{M}(\mathbf{r})$ fulfill conditions $1) - 3)$. Then we know that the divisor $m(x) \in \mathcal{L}_q(x, q^m)$ has q-degree less than k and it holds $N(x) = D(m(x))$. Since it is in $\mathfrak{M}(\mathbf{r})$ we know by Theorem 19 that $D(m(g_i)) - D(r_i) = 0$ for all i. Define $\mathbf{c} := (m(g_1) \ldots m(g_n))$, then it follows from Theorem 5 that $d_R(\mathbf{c}, \mathbf{r}) = t$.

Therefore, list decoding within rank radius t is equivalent to finding all elements $[N(x) \quad -D(x)]$ in $\mathfrak{M}(\mathbf{r})$ with $(0, k-1)$-weighted q-degree less than $t+k$ and leading position 2, such that $N(x)$ is symbolically divisible on the left by $D(x)$. Note that this is a generalization of the interpolation-based decoding method from [13]. The difference is that our method can also decode beyond the unique decoding radius.

We can now describe the list decoding algorithm. Since in most applications one wants to find the set of all closest codewords to the received word, our algorithm will do exactly this. In contrast, a complete list decoder with a prescribed radius t finds all codewords within radius t from the received word, even if some of them are closer than others.

Algorithm 1 describes the decoding algorithm. It will iteratively search for all elements in $\mathfrak{M}(\mathbf{r})$ of $(0, k-1)$-weighted q-degree $t + k - 1$ for increasing t and check the requirements of Theorem 20. As soon as solutions are found, t will not be increased and the algorithm terminates.

As in Section 3 we use the notation $f = [f_1(x) \quad f_2(x)], b^{(i)} = [b_1^{(i)}(x) \quad b_2^{(i)}(x)]$, etc. for elements of the interpolation module $\mathfrak{M}(\mathbf{r})$.

Theorem 21 *Algorithm 1 yields a list of all message polynomials such that the corresponding codeword is closest to the received word.*

Proof We will prove this in two steps; first we show that any closest codeword will be in the output list, then we show that any element in the output list of the algorithm is a closest codeword.

For the first direction let $\mathbf{c} \in \mathbb{F}_{q^m}^n$ be a closest codeword with corresponding message polynomial $m(x) \in \mathcal{L}_q(x, q^m)_{<k}$ and $t := d_R(\mathbf{c}, \mathbf{r})$. We know from Theorem

Algorithm 1 Minimal list decoding of a Gabidulin code of dimension k with generators $g_1, \ldots, g_n \in \mathbb{F}_{q^m}^n$.

Require: Received word $\mathbf{r} \in \mathbb{F}_{\mathbf{q^m}}^{\mathbf{n}}$.

1. Compute $\Pi(x)$ and $\Lambda_{\mathbf{g},\mathbf{r}}(x)$, both in $\mathcal{L}_q(x, q^m)$. Define the interpolation module

$$\mathfrak{M}(\mathbf{r}) := \mathrm{rs} \begin{bmatrix} \Pi(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

2. Compute a minimal basis $B = \{b^{(1)}, b^{(2)}\}$ of $\mathfrak{M}(\mathbf{r})$ with respect to the $(0, k-1)$-weighted degree, with $\mathrm{lpos}(b^{(1)}) = 1$ and $\mathrm{lpos}(b^{(2)}) = 2$.
3. Define ℓ_1, ℓ_2 as the $(0, k-1)$-weighted q-degrees of $b^{(1)}, b^{(2)}$, respectively.
4. Define **list**:= [ ] (an empty list) and $j := 0$.

```
while list= [ ] do
  for all β(x) ∈ L_q(x, q^m), qdeg(a(x)) ≤ ℓ_2 − ℓ_1 + j do
    for all monic γ(x) ∈ L_q(x, q^m), qdeg(b(x)) = j do
      f := β(x) ∘ b^(1) + γ(x) ∘ b^(2)
      if there exists m(x) ∈ L_q(x, q^m) such that f_1(x) = f_2(m(x)) then
        add m(x) to list
      end if
    end for
  end for
  j := j + 1
end while
return list
```

20 that there exists $f = [f_1(x) \quad f_2(x)] = [f_2(m(x)) \quad f_2(x)] \in \mathfrak{M}(\mathbf{r})$, with $\mathrm{lpos}(f) = 2$ and $\mathrm{qdeg}(f_2(x)) = t$. I.e. there exist $\beta(x), \gamma(x) \in \mathcal{L}_q(x, q^m)$ such that $\beta(x) \circ b^{(1)} + \gamma(x) \circ b^{(2)} = f$. Furthermore, we know from the PLM property (Theorem 16), that

$$\mathrm{lm}(f) = \max\{\mathrm{lm}(\beta(x)) \circ \mathrm{lm}(b^{(1)}), \mathrm{lm}(\gamma(x)) \circ \mathrm{lm}(b^{(2)})\}$$

which implies that $\mathrm{lm}(f) = \mathrm{lm}(\gamma(x)) \circ \mathrm{lm}(b^{(2)})$, since $\mathrm{lpos}(b^{(i)}) = i$ for $i = 1, 2$. Therefore we get

$$t = \mathrm{qdeg}(\gamma(x)) + (\ell_2 - k + 1) \iff \mathrm{qdeg}(\gamma(x)) = t - (\ell_2 - k + 1).$$

Because of the leading position we furthermore get

$$\mathrm{qdeg}(\beta(x)) + \mathrm{qdeg}(b_1^{(1)}(x)) \leq \mathrm{qdeg}(\gamma(x)) + \mathrm{qdeg}(b_2^{(2)}(x)) + k - 1$$

$$\iff \mathrm{qdeg}(\beta(x)) \leq \mathrm{qdeg}(\gamma(x)) + \ell_2 - \ell_1.$$

Set $j := t - \ell_2 + k - 1$, then the above conditions translate into $\mathrm{qdeg}(\gamma(x)) = j$ and $\mathrm{qdeg}(\beta(x)) \leq j + \ell_2 - \ell_1$, which is exactly the parametrization used in the algorithm. We can choose $\gamma(x)$ monic, since the non-zero scalar coefficients of our polynomials are from $\mathbb{F}_{q^m}$ and are thus invertible.

Note that increasing j by one is equivalent to increasing t by one. Therefore, if t is the minimal distance between any codeword and the received word, the algorithm will not terminate at any $j < t - \ell_2 + k - 1$. Thus, we have shown that the algorithm will produce $m(x)$ as an output element.

For the other direction let $m(x)$ be an element of the output list of Algorithm 1. Then there exists $f = [f_1(x) \quad f_2(x)] = [f_2(m(x)) \quad f_2(x)] \in \mathfrak{M}(\mathbf{r})$. Furthermore there exist $j \in \mathbb{N}_0$ and $\beta(x), \gamma(x) \in \mathcal{L}_q(x, q^m)$ with $\mathrm{qdeg}(\beta(x)) \leq$

$\ell_2 - \ell_1 + j, \text{qdeg}(\gamma(x)) = j$ such that $\beta(x) \circ b^{(1)} + \gamma(x) \circ b^{(2)} = f$. Since $\text{qdeg}(b_1^{(1)}) = \ell_1, \text{qdeg}(b_2^{(1)}) < \ell_1 - k + 1$ and $\text{qdeg}(b_1^{(2)}) \leq \ell_2, \text{qdeg}(b_2^{(2)}) < \ell_2 - k + 1$, it follows that

$$\text{qdeg}(f_1(x)) \leq \ell_2 + j \quad , \quad \text{qdeg}(f_2(x)) = \ell_2 + j - k + 1.$$

Hence all the requirements of Theorem 20 are fulfilled for $t = \ell_2 + j - k + 1$ and $m(x)$ belongs to a codeword $\mathbf{c} = (m(g_1) \dots m(g_n))$ with $d_R(\mathbf{c}, \mathbf{r}) = \ell_2 + j - k + 1$. It remains to show that there is no closer codeword. Assume there exists $m'(x) \in \mathcal{L}_q(x, q^m)$ of q-degree less than k with corresponding codeword $\mathbf{c}'$ such that $t' := d_R(\mathbf{c}', r) < \ell_2 + j - k + 1$. Then, by the first part of this proof, the algorithm would terminate at $j = t' - \ell_2 + k - 1 < t - \ell_2 + k - 1$ and would not produce $m(x)$ as an output element.

It remains to show that there are no codewords at rank distance less than $\ell_2 - k + 1$, since this is the distance for the initial loop with $j = 0$. Assume there would be such a codeword with corresponding message polynomial $m(x) \in \mathcal{L}_q(x, q^m)$. Then there exists $D(x) \in \mathcal{L}_q(x, q^m)$ with q-degree less than $\ell_2 - k + 1$ such that $f' := [D(m(x)) \;\; D(x)]$ is in $\mathfrak{M}(\mathbf{r})$. Then the $(0, k-1)$-weighted q-degree of f' is less than ℓ_2, which means that B is not a minimal basis of $\mathfrak{M}(\mathbf{r})$, which is a contradiction.

4.2 Construction of a Minimal Basis

We will now explain two different ways of obtaining the minimal basis for the interpolation module, as required in Algorithm 1. The first one will use the extended Euclidean algorithm for composition while the second one is an iterative algorithm. Note that unique decoding with the Euclidean algorithm was done in a Gao-like algorithm in [26]. Similarly, unique decoding (possibly beyond half the minimum distance as a list-1 decoder) with an iterative algorithm analogous to our Algorithm 3 can be found in [6,13,28]. Our algorithm differs from these works in the sense that we compute all (and not only one) closest codewords to the received word. For this our set-up in terms of modules and particularly our parametrization result are novel ingredients that we believe give new insights into this topic.

We will first describe an algorithm to find the minimal basis of $\mathfrak{M}(\mathbf{r})$ with the help of the extended Euclidean algorithm (EEA) for q-linearized polynomials with respect to composition. This variant of the Euclidean algorithm is well-known, see e.g. [26, Algorithm 2.3], and works analogously to the classical EEA for normal polynomials. Since we need to distinguish left and right symbolic division, there exists a right and left EEA in $\mathcal{L}_q(x, q^m)$. In this work we are only interested in the right EEA. For given $f(x), g(x) \in \mathcal{L}_q(x, q^m)$ with $\text{qdeg}(f(x)) \geq \text{qdeg}(g(x))$ this algorithm computes $q(x), r(x) \in \mathcal{L}_q(x, q^m)$ such that $\text{qdeg}(r(x)) < \text{qdeg}(g(x))$ and

$$f(x) = q(x) \circ g(x) + r(x).$$

Theorem 22 *Algorithm 2 produces a minimal basis $B = \{b^{(1)}, b^{(2)}\}$ for our interpolation module $\mathfrak{M}(\mathbf{r})$. Moreover, $\text{lpos}(b^{(1)}) = 1$ and $\text{lpos}(b^{(2)}) = 2$.*

Proof As shown in Theorem 19, for $j = 0$ we have a basis of $\mathfrak{M}(\mathbf{r})$. Then for $j \geq 1$ we have that

$$\begin{bmatrix} P_j(x) & K_j(x) \\ N_j(x) & D_j(x) \end{bmatrix} = \begin{bmatrix} 0 & x \\ x & -q_{j-1}(x) \end{bmatrix} \circ \begin{bmatrix} P_{j-1}(x) & K_{j-1}(x) \\ N_{j-1}(x) & D_{j-1}(x) \end{bmatrix}$$

$$\iff \begin{bmatrix} q_{j-1}(x) & x \\ x & 0 \end{bmatrix} \circ \begin{bmatrix} P_j(x) & K_j(x) \\ N_j(x) & D_j(x) \end{bmatrix} = \begin{bmatrix} P_{j-1}(x) & K_{j-1}(x) \\ N_{j-1}(x) & D_{j-1}(x) \end{bmatrix},$$

hence also the new matrix is a basis of $\mathfrak{M}(\mathbf{r})$.

We know that at level 0 both basis vectors have leading position 1. The **while**-condition assures that the algorithm terminates as soon as we have a basis where the second vector has leading position 2, while the first still has leading position 1. This automatically implies that this basis is minimal, according to Proposition 15.

Example 23 *Consider the Gabidulin code in* $\mathbb{F}_{2^3} \cong \mathbb{F}_2[\alpha]$ *(with* $\alpha^3 = \alpha + 1$*) of dimension* $k = 2$ *with generator matrix*

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

and the received word

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

Then we construct the interpolation module

$$M(\mathbf{r}) = \mathrm{rs} \begin{bmatrix} \Pi(x) & 0 \\ -\Lambda_{\mathbf{r}}(x) & x \end{bmatrix} = \mathrm{rs} \begin{bmatrix} x^8 + x & 0 \\ \alpha^2 x^4 + \alpha^5 x & x \end{bmatrix}.$$

To compute a minimal basis we use the Euclidean algorithm and get

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

Algorithm 2 Computation of minimal basis of $\mathfrak{M}(\mathbf{r})$ via the extended Euclidean algorithm.

Require: Received word $\mathbf{r}$; polynomials $\Pi(x)$ and $\Lambda_{\mathbf{g},\mathbf{r}}(x)$.
Initialize $j = 0$ and define $P_0(x), N_0(x), K_0(x), D_0(x)$ as

$$\begin{bmatrix} P_0(x) & K_0(x) \\ N_0(x) & D_0(x) \end{bmatrix} := \begin{bmatrix} \Pi(x) & 0 \\ -\Lambda_{\mathbf{g},\mathbf{r}}(x) & x \end{bmatrix}.$$

while $\mathrm{qdeg}(D_j) + k - 1 < \mathrm{qdeg}(N_j)$ **do**
Apply the EEA to compute $q_j(x), r_j(x) \in \mathcal{L}_q(x, q^m)$ such that $P_j(x) = q_j(x) \circ N_j(x) + r_j(x)$ and $\mathrm{qdeg}(r_j) < \mathrm{qdeg}(N_j)$.
Update

$$\begin{bmatrix} P_{j+1}(x) & K_{j+1}(x) \\ N_{j+1}(x) & D_{j+1}(x) \end{bmatrix} := \begin{bmatrix} N_j(x) & D_j(x) \\ r_j(x) & K_j(x) - q_j(x) \circ D_j(x) \end{bmatrix}.$$

Set $j := j + 1$.
end while
return $b^{(1)} := [\, P_j(x) \quad K_j(x) \,]$ and $b^{(2)} := [\, N_j(x) \quad D_j(x) \,]$

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*

$$\begin{bmatrix} g_1^{(1)} & g_1^{(2)} \\ g_2^{(1)} & g_2^{(2)} \end{bmatrix} = \begin{bmatrix} \alpha^2 x^4 + \alpha^5 x & x \\ \alpha^6 x^2 + x & \alpha^3 x^2 \end{bmatrix}.$$

Hence we get $\ell_1 = 2$ *and* $\ell_2 = 2$*, i.e. we want to use all* $\beta(x) \in \mathcal{L}_2(x, 2^3)$ *with* 2*-degree less than or equal to* 0 *and all monic* $\gamma(x) \in \mathcal{L}_2(x, 2^3)$ *with* 2*-degree equal to* 0*. Thus,* $\beta(x) = b_0 x$ *for* $b_0 \in \mathbb{F}_{2^3}$ *and* $\gamma(x) = x$*. We get divisibility for* $b_0 \in \mathbb{F}_{2^3} \backslash \{0\}$*. The corresponding message polynomials and codewords are*

$$m_1(x) = x^2 + \alpha x \quad , \quad c_1 = (\,\alpha^3\ 1\ \alpha^3),$$

$$m_2(x) = \alpha^5 x^2 + \alpha^2 x \quad , \quad c_1 = (\,\alpha^3\ \alpha\ \alpha),$$

$$m_3(x) = \alpha^3 x^2 + \alpha^4 x \quad , \quad c_1 = (\,\alpha^2 + 1\ 0\ \alpha^2),$$

$$m_4(x) = \alpha^4 x^2 \quad , \quad c_3 = (\,\alpha^2 + \alpha\ \alpha^2 + 1\ \alpha),$$

$$m_5(x) = \alpha^6 x^2 + \alpha^6 x \quad , \quad c_1 = (\,0\ \alpha^3\ 1),$$

$$m_6(x) = \alpha x^2 + x \quad , \quad c_2 = (\,\alpha^3\ 1\ \alpha^3).$$

All these codewords are rank distance 1 *away from* $\mathbf{r}$*. Note that their Hamming distance to* $\mathbf{r}$ *varies from* 1 *to* 3*.*

We will now derive an iterative algorithm for the computation of a minimal basis of the interpolation module. For this we need the following result, which was first stated in our recent paper [9].

Lemma 24 *For* $i = 1, \ldots, n$ *denote by* $\mathfrak{M}_i$ *the interpolation module for* $(g_1, \ldots, g_i)$ *and* $(r_1, \ldots, r_i)$*. Let*

$$\begin{bmatrix} P(x) & -K(x) \\ N(x) & -D(x) \end{bmatrix}$$

be a basis for $\mathfrak{M}_{i-1}$ *and*

$$\Gamma_i := P(g_i) - K(r_i) \quad , \quad \Delta_i := N(g_i) - D(r_i).$$

If $\Gamma_i \neq 0$*, then the row vectors of*

$$\begin{bmatrix} b^{(1)} \\ b^{(2)} \end{bmatrix} := \begin{bmatrix} x^q - \Gamma_i^{q-1} x & 0 \\ \Delta_i x & -\Gamma_i x \end{bmatrix} \circ \begin{bmatrix} P(x) & -K(x) \\ N(x) & -D(x) \end{bmatrix}$$

form a basis of $\mathfrak{M}_i$*. If* $\Delta_i \neq 0$*, then the row vectors of*

$$\begin{bmatrix} \Delta_i x & -\Gamma_i x \\ 0 & x^q - \Delta_i^{q-1} x \end{bmatrix} \circ \begin{bmatrix} P(x) & -K(x) \\ N(x) & -D(x) \end{bmatrix}$$

form a basis of $\mathfrak{M}_i$*.*

Proof We first consider the first case and show that both $b^{(1)}$ and $b^{(2)}$ are in $\mathfrak{M}_i$. From the assumptions it follows that $P(g_j) = K(r_j)$ and that $N(g_j) = D(r_j)$ for $1 \leq j < i$. Moreover, the two entries of $b^{(1)}$ are given by

$$(x^q - \Gamma_i^{q-1}x) \circ P(x) = P(x)^q - \Gamma_i^{q-1}P(x),$$

$$(x^q - \Gamma_i^{q-1}x) \circ K(x) = K(x)^q - \Gamma_i^{q-1}K(x),$$

thus $P(g_j)^q - \Gamma_i^{q-1}P(g_j) - K(r_j)^q + \Gamma_i^{q-1}K(r_j) = 0$ for $1 \leq j \leq i$. For $b^{(2)}$ we get

$$\Delta_i P(g_j) - \Gamma_i N(g_j) - \Delta_i K(r_j) + \Gamma_i D(r_j) =$$

$$\Delta_i(P(g_j) - K(r_j)) - \Gamma_i(N(g_j) - D(r_j)) = \Delta_i\Gamma_i - \Gamma_i\Delta_i = 0$$

for $1 \leq j \leq i$. Thus, $b^{(1)}$ and $b^{(2)}$ are elements of $\mathfrak{M}_i$.

It remains to show that $b^{(1)}$ and $b^{(2)}$ span the whole interpolation module (and not just a submodule of it). For this, it is sufficient to show that $[\ \Pi_{i-1}(x) \quad 0\ ]$ and $[\ \Lambda_{i-1}(x) \quad -x\ ]$ are linear combinations of $b^{(1)}$ and $b^{(2)}$. Since $[\ P(x) \quad -K(x)\ ]$ and $[\ N(x) \quad -D(x)\ ]$ form a basis of $\mathfrak{M}_i$, there exist $\bar{\beta}(x), \bar{\gamma}(x) \in \mathcal{L}_q(x, q^m)$ such that

$$\bar{\beta}(x) \circ [\ P(x) \quad -K(x)\ ] + \bar{\gamma}(x) \circ [\ N(x) \quad -D(x)\ ] = [\ \Pi_{i-1}(x) \quad 0\ ].$$

Let $\beta(x), \gamma(x) \in \mathcal{L}_q(x, q^m)$ be such that

$$\beta(x) \circ (x^q - \Gamma_i^{q-1}x) = (x^q - \Pi_{i-1}(g_i)^{q-1}x) \circ \left(\bar{\gamma}(\frac{\Delta_i}{\Gamma_i}x) + \bar{\beta}(x)\right),$$

$$\gamma(x) = -(x^q - \Pi_{i-1}(g_i)^{q-1}x) \circ \bar{\gamma}(\frac{1}{\Gamma_i}x).$$

Note that it can easily be checked that Γ_i is a root of the right side of the previous equation, thus $\beta(x)$ is well-defined by Lemma 4 . Denote the first and second row of the new basis by $b^{(1)}$ and $b^{(2)}$, respectively. Then

$$\begin{aligned}
\beta(x) \circ b_1^{(1)} + \gamma(x) \circ b_1^{(2)} &= \beta(x) \circ (x^q - \Gamma_i^{q-1}x) \circ P(x) + \gamma(x) \circ (\Delta_i P(x) - \Gamma_i N(x)) \\
&= (x^q - \Pi_{i-1}(g_i)^{q-1}x) \circ \left(\bar{\gamma}(\frac{\Delta_i}{\Gamma_i}x) + \bar{\beta}(x)\right) \circ P(x) + \gamma(\Delta_i P(x)) - \gamma(\Gamma_i N(x)) \\
&= \left((x^q - \Pi_{i-1}(g_i)^{q-1}x) \circ \bar{\beta}(x) - \gamma(\Delta_i x)\right) \circ P(x) + \gamma(\Delta_i P(x)) - \gamma(\Gamma_i N(x)) \\
&= \left((x^q - \Pi_{i-1}(g_i)^{q-1}x) \circ \bar{\beta}(x)\right) \circ P(x) + (x^q - \Pi_{i-1}(g_i)^{q-1}x) \circ \bar{\gamma}(N(x)) \\
&= (x^q - \Pi_{i-1}(g_i)^{q-1}x) \circ \left(\bar{\beta}(P(x)) + \bar{\gamma}(N(x))\right) \\
&= (x^q - \Pi_{i-1}(g_i)^{q-1}x) \circ \Pi_{i-1}(x) = \Pi_i(x),
\end{aligned}$$

and

$$\begin{aligned}
\beta(x) \circ b_2^{(1)} + \gamma(x) \circ b_2^{(2)} &= -\beta(x) \circ (x^q - \Gamma_i^{q-1}x) \circ K(x) - \gamma(x) \circ (\Delta_i K(x) - \Gamma_i D(x)) \\
&= \left((x^q - \Pi_{i-1}(g_i)^{q-1}x) \circ \bar{\beta}(x) - \gamma(\Delta_i x)\right) \circ K(x) + \gamma(\Delta_i K(x)) - \gamma(\Gamma_i D(x)) \\
&= (x^q - \Pi_{i-1}(g_i)^{q-1}x) \circ \left(\bar{\beta}(K(x)) + \bar{\gamma}(D(x))\right) = 0.
\end{aligned}$$

Thus $\beta(x) \circ b^{(1)} + \gamma(x) \circ b^{(2)} = [\ \Pi_i(x) \quad 0\ ]$, i.e. $[\ \ \Pi_i(x) \quad 0\ ]$ is in the module spanned by the new basis.

Analogously, if we have that $\bar{c}(x) \circ [\ P(x) \quad -K(x)\ ] + \bar{d}(x) \circ [\ N(x) \quad -D(x)\ ] = [\ \Lambda_{i-1}(x) \quad -x\ ]$ and define $c(x), d(x) \in \mathcal{L}_q(x, q^m)$ such that

$$d(x) = -\left(\bar{d}(x) + \frac{\Lambda_{i-1}(g_i) - r_i}{\Pi_{i-1}(g_i)}\bar{\gamma}(x)\right) \circ (\frac{1}{\Gamma_i}x)$$

$$c(x) \circ (x^q - \Gamma_i^{q-1}x) = \bar{c}(x) + \frac{\Lambda_{i-1}(g_i) - r_i}{\Pi_{i-1}(g_i)}\bar{\beta}(x) - d(\Delta_i x)$$

we get

$$c(x) \circ b^{(1)} + d(x) \circ b^{(2)} = [\ \Lambda_i(x) \quad -x\ ].$$

Hence, we have shown that the new basis $\{b^{(1)}, b^{(2)}\}$ spans the whole interpolation module.

For the second case note that

$$\mathrm{rs}\left(\begin{bmatrix} \Delta_i x & -\Gamma_i x \\ 0 & x^q - \Delta_i^{q-1}x \end{bmatrix} \circ \begin{bmatrix} P(x) & -K(x) \\ N(x) & -D(x) \end{bmatrix}\right)$$

$$= \mathrm{rs}\left(\begin{bmatrix} x^q - \Delta_i^{q-1}x & 0 \\ \Gamma_i x & -\Delta_i x \end{bmatrix} \circ \begin{bmatrix} N(x) & -D(x) \\ P(x) & -K(x) \end{bmatrix}\right),$$

which corresponds to the first case after exchanging $P(x)$ with $N(x)$ and $K(x)$ with $D(x)$ (and vice versa).

Remark 25 *In the notation of Proposition 3, applying the previous theorem to $P(x) = \Pi_{i-1}(x), K(x) = 0, N(x) = \Lambda_{i-1}(x)$ and $D(x) = -x$, leads to a computation that is identical up to a constant to the one in Proposition 3 in which the q-annihilator polynomial and the q-Lagrange polynomial are iteratively constructed.*

Using Lemma 24 as our main ingredient, we now set out to design an iterative algorithm that computes a minimal basis for $\mathfrak{M}_i$ at each step i.

Theorem 26 *Algorithm 3 yields a minimal basis of the interpolation module $\mathfrak{M}(\mathbf{r})$, where the leading position of the first row is 1 and the leading position of the second row is 2.*

Proof Denote by M_1 the matrix we multiply by on the left in the first IF statement and by M_2 the one in the ELSE statement of the algorithm. We know from Lemma 24 that at each step, B_i is a basis for the interpolation module $\mathfrak{M}_i$. We now show that it is a minimal basis with respect to the $(0, k-1)$-weighted term-over-position monomial order via induction on i. Assume that at step i the first row has leading position 1 and the second row has leading position 2, i.e. $\mathrm{qdeg}(P_i(x)) > \mathrm{qdeg}(K_i(x)) + k - 1$ and $\mathrm{qdeg}(N_i(x)) \leq \mathrm{qdeg}(D_i(x)) + k - 1$. If $\mathrm{qdeg}(P_i(x)) \leq \mathrm{qdeg}(D_i(x)) + k - 1$ we composite on the left by M_1. Hence,

$$\mathrm{qdeg}(P_{i+1}(x)) = \mathrm{qdeg}(P_i(x)) + 1$$

and

$$\mathrm{qdeg}(K_{i+1}(x)) = \mathrm{qdeg}(K_i(x)) + 1 < \mathrm{qdeg}(P_i(x)) - k + 2 = \mathrm{qdeg}(P_{i+1}(x)) - k + 1.$$

Algorithm 3 Iterative computation of minimal basis of $\mathfrak{M}(\mathbf{r})$.

Require: Linearly independent $g_1, \ldots, g_n \in \mathbb{F}_{q^m}$, received $r_1, \ldots, r_n \in \mathbb{F}_{q^m}$.

Initialize **list**:= [ ], $j := 0$, $B_0 := \begin{bmatrix} x & 0 \\ 0 & x \end{bmatrix}$.

We denote $B_i := \begin{bmatrix} P_i(x) & -K_i(x) \\ N_i(x) & -D_i(x) \end{bmatrix}$.

for i from 1 to n **do**

$\Gamma_i := P_{i-1}(g_i) - K_{i-1}(r_i) \quad , \quad \Delta_i := N_{i-1}(g_i) - D_{i-1}(r_i)$.

if $[\mathrm{qdeg}(P_{i-1}(x)) \leq \mathrm{qdeg}(D_{i-1}(x)) + k - 1$ **and** $\Gamma_i \neq 0]$ or $\Delta_i = 0$ **then**

$$B_i := \begin{bmatrix} x^q - \Gamma_i^{q-1} x & 0 \\ \Delta_i x & -\Gamma_i x \end{bmatrix} \circ B_{i-1}$$

else

$$B_i := \begin{bmatrix} \Delta_i x & -\Gamma_i x \\ 0 & x^q - \Delta_i^{q-1} x \end{bmatrix} \circ B_{i-1}$$

end if

end for

return B_n

Thus, the leading position of the first row of B_{i+1} is still 1. Moreover,

$$\mathrm{qdeg}(N_{i+1}(x)) \leq \max\{\mathrm{qdeg}(P_i(x)), \mathrm{qdeg}(N_i(x))\} \leq \mathrm{qdeg}(D_i(x)) + k - 1$$

and, since the assumptions imply that $\mathrm{qdeg}(K_i(x)) < \mathrm{qdeg}(D_i(x))$,

$$\mathrm{qdeg}(D_{i+1}(x)) = \max\{\mathrm{qdeg}(K_i(x)), \mathrm{qdeg}(D_i(x))\} = \mathrm{qdeg}(D_i(x)).$$

Thus the leading position of the second row is 2. Since the assumptions are true for B_0 the statement follows via induction.

Analogously one can prove that composition with M_2 yields a basis of $\mathfrak{M}_i$ with different leading positions in the two rows. I.e. at each step we get a basis of $\mathfrak{M}_i$ with different leading positions, which is by Proposition 15 a minimal basis. Thus, after n steps, B_n is a minimal basis for the interpolation module $\mathfrak{M}(\mathbf{r})$.

Remark 27 *It can be verified that, due to the linear independence of $g_1, \ldots, g_k$, the first k steps of the algorithm coincide up to a constant with the computation in Proposition 3. In other words, up to a constant, at step k the algorithm has computed the q-annihilator polynomial and the q-Lagrange polynomial corresponding to the data so far.*

Example 28 *Consider the same setting as in Example 23, i.e. a Gabidulin code in $\mathbb{F}_{2^3} \cong \mathbb{F}_2[\alpha]$ (with $\alpha^3 = \alpha + 1$) with generator matrix*

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

and the received word $\mathbf{r} = (\, \alpha^3 \; 0 \; \alpha \,)$. We iteratively compute

$$B_1 = \begin{bmatrix} x^2 + x & 0 \\ (\alpha + 1)x & x \end{bmatrix},$$

$$B_2 = \begin{bmatrix} x^4 + (\alpha^2 + \alpha + 1)x^2 + (\alpha^2 + \alpha)x & 0 \\ (\alpha^2 + \alpha)x^2 + (\alpha^2 + \alpha + 1)x & (\alpha^2 + \alpha)x \end{bmatrix},$$

$$B_3 = \begin{bmatrix} \alpha^2 x^4 + \alpha^5 x & x \\ \alpha x^4 + \alpha^4 x^2 + x & \alpha x^2 + \alpha^6 x \end{bmatrix}.$$

B_3 is a minimal $(0,1)$-weighted basis of the interpolation module. We get $\ell_1 = 2$ and $\ell_2 = 2$, i.e. we want to use all $\beta(x) \in \mathcal{L}_2(x, 2^3)$ with 2-degree less than or equal to 0 and all monic $\gamma(x) \in \mathcal{L}_2(x, 2^3)$ with 2-degree equal to 0. Thus, $\beta(x) = b_0 x$ for $b_0 \in \mathbb{F}_{2^3}$ and $\gamma(x) = x$. We get divisibility for $b_0 \in \mathbb{F}_{2^3} \backslash \{\alpha^6\}$. The corresponding message polynomials are indeed the same as the ones from Example 23, although the minimal basis of the interpolation module differs from the one in Example 23. This can also be verified by the fact that

$$\begin{bmatrix} \alpha^2 x^4 + \alpha^5 x & x \\ \alpha^6 x^2 + x & \alpha^3 x^2 \end{bmatrix} = \begin{bmatrix} x & 0 \\ \alpha x & \alpha^2 x \end{bmatrix} \circ \begin{bmatrix} \alpha^2 x^4 + \alpha^5 x & x \\ \alpha x^4 + \alpha^4 x^2 + x & \alpha x^2 + \alpha^6 x \end{bmatrix}$$

$$\implies [\ \beta'(x) \quad x\ ] \circ \begin{bmatrix} \alpha^2 x^4 + \alpha^5 x & x \\ \alpha^6 x^2 + x & \alpha^3 x^2 \end{bmatrix} =$$

$$\alpha^2 [\ \alpha^5 \beta'(x) + \alpha^6 x \quad x\ ] \circ \begin{bmatrix} \alpha^2 x^4 + \alpha^5 x & x \\ \alpha x^4 + \alpha^4 x^2 + x & \alpha x^2 + \alpha^6 x \end{bmatrix},$$

which implies that $\beta'(x)$ in the setting of Example 23 corresponds to $\beta(x) = \alpha^5 \beta'(x) + \alpha^6 x$ in this example.

We can now prove the following properties of the two elements of a minimal basis of the interpolation module.

Lemma 29 *Consider a Gabidulin code $C \subseteq \mathbb{F}_{q^m}^n$ of dimension k. Let $\mathfrak{M}(\mathbf{r})$ be the interpolation module of the received word $\mathbf{r} \in \mathbb{F}_{q^m}^n$ with minimal basis $B = \{b^{(1)}, b^{(2)}\}$ where $\operatorname{lpos}(b^{(i)}) = i$ for $i = 1, 2$. Furthermore denote by ℓ_i be the $(0, k-1)$-weighted q-degree of $b^{(i)}$ for $i = 1, 2$. Then*

$$\ell_1 + \ell_2 = n + k - 1$$

or equivalently

$$\operatorname{qdeg}(b_1^{(1)}) + \operatorname{qdeg}(b_2^{(2)}) = n.$$

Proof By Proposition 17 we know that the q-degrees of any minimal basis of the interpolation module $\mathfrak{M}_{\mathbf{r}}$ have to add up to the same number, hence it is enough to show that they add up to $n + k - 1$ for one particular basis. Consider the iterative construction of a minimal basis from Algorithm 3. It is easy to see that the initial basis has weighted q-degrees 0 and $k - 1$. Moreover, at each step the q-degree of one row is increased by one, whereas the q-degree of the other row remains the same. Thus, the sum of the two q-degrees is increased by 1 at each step. Since we get the desired basis of $\mathfrak{M}_{\mathbf{r}}$ at the n-th step, the statement follows.

We will now show that in the case where the received word is within the unique decoding radius the algorithm only computes the minimal basis of the interpolation module and then performs one symbolic division to find the message polynomial.

Lemma 30 *In the setting of Lemma 29 let $t := \min\{d_R(\mathbf{c}, \mathbf{r}) \mid \mathbf{c} \in C\}$. Then*

$$\ell_2 \leq t + k - 1$$

or equivalently $\operatorname{qdeg}(b_2^{(2)}) \leq t$ *. Furthermore,*

$$\ell_1 = \operatorname{qdeg}(b_1^{(1)}) \geq n - t.$$

Proof Let $m(x) \in \mathcal{L}_q(x, q^m)$ be the message polynomial corresponding to the codeword $\mathbf{c}$. Then by Theorem 20, there exist $D(x) \in \mathcal{L}_q(x, q^m)$ of q-degree t such that $f := [\, D(m(x)) \;\; D(x)\,]$ is an element of the interpolation module with leading position 2. By the PLM property from Theorem 16 we know that $\mathrm{lm}(f) = \mathrm{lm}(a(x) \circ b^{(2)})$ for some $a(x) \in \mathcal{L}_q(x, q^m)$, i.e. $\mathrm{lm}(f) \geq \mathrm{lm}(b^{(2)})$, which implies the first statement since the leading positions of both elements are 2. We know from Lemma 29 that $\ell_1 + \ell_2 = n + k - 1$, i.e.

$$\ell_1 = n + k - 1 - \ell_2 \geq n - t.$$

Lemma 31 *In the previous setting, if* $t \leq (n-k)/2$*, then* $\ell_2 = t+k-1$*, or equivalently* $\mathrm{qdeg}(b_2^{(2)}) = t$*, and* $\ell_1 = n - t$*.*

Proof We know from Lemma 30 that

$$\ell_1 \geq n - t \geq \frac{n+k}{2}.$$

Since the vector corresponding to the closest codeword has $(0, k-1)$-weighted q-degree $t + k - 1 < \ell_1$, this vector is $a(x) \circ b^{(2)}$ for some $a(x) \in \mathcal{L}_q(x, q^m)$. But if the divisibility requirement is fulfilled for $a(x) \circ b^{(2)}$, then it must also be fulfilled for $b^{(2)}$. Hence $b^{(2)}$ must correspond to the closest codeword and is thus of weighted q-degree $t + k - 1$ (by Theorem 20). Note that the last step could also be justified by using the fact that we are inside the unique decoding radius. From Lemma 29 we then get that $\ell_1 = n + k - 1 - \ell_2 = n - t$.

Corollary 32 *It follows that, if the received word is within the unique decoding radius, Algorithm 1 only performs one loop and hence only one symbolic division to find the message polynomial correspoding to the unique closes codeword.*

4.3 Complexity Analysis

We will now analyze the computational complexity of the previous algorithms. We will start with the easier task of analyzing Algorithms 2 and 3. Then we will derive some results on the degrees of the minimal basis elements that will help us derive the overall decoding complexity in terms of the rank distance of the closest codeword to the received word.

For the whole subsection we will use the notation of the previous subsection, i.e. $b^{(1)}, b^{(2)} \in \mathcal{L}_q(x, q^m)$ form a minimal basis of the interpolation module $\mathfrak{M}(\mathbf{r})$ and ℓ_i is the $(0, k-1)$-weighted q-degree of $b^{(i)}$ for $i = 1, 2$. Moreover, $\mathrm{lpos}(b^{(i)}) = i$ for $i = 1, 2$. $\Pi_{\mathbf{g}}(x)$ is the annihilator polynomial for the generators of the code $g_1, \ldots, g_n \in \mathbb{F}_{q^m}$ and $\Lambda_{\mathbf{r},\mathbf{g}}(x)$ is the q-Lagrange polynomial with respect to the received vector $\mathbf{r} \in \mathbb{F}_{\mathbf{q^m}}^{\mathbf{n}}$.

Lemma 33 $\Pi_{\mathbf{g}}(x)$ *can be computed with at most* $O_{q^m}(qn^2)$ *operations.*

Proof Consider the iterative construction from Proposition 3. At step i we need to compute $\Pi_i(x) = (x^q - g_i^{q-1}x) \circ \Pi_{i-1}(x)$. We compute a $(q-1)$-th power of g_i, which are at most $q - 2$ multiplications. Moreover, we need to multiply $\Pi_{i-1}(x)$ with this power, which needs at most i operations since $\mathrm{qdeg}(\Pi_{i-1}(x)) = i - 1$. Similarly we need to take the q-th power of all terms of $\Pi_{i-1}(x)$, which needs

at most qi operations. The last step is to take the difference of the two resulting polynomials (where one has q-degree i and the other $i+1$), hence at most $i+1$ operations. Since i is upper bounded by n, we get an upper bound of $O_{q^m}(qn)$ operations at each step. Since there are n steps the overall complexity is upper bounded by $O_{q^m}(qn^2)$.

Note that if we use a normal basis of $\mathbb{F}_{q^m}$, taking q-th powers becomes negligible (see e.g. [26, Section 3.1]). Then the above complexity order reduces to $O_{q^m}(n^2)$.

Remark 34 *Since $\Pi_{\mathbf{g}}(x)$ does not depend on the received word, we can precompute and store it. In the following we assume that we precomputed $\Pi_{g_1,\dots,g_i}(x)$ for $i=1,\dots,n$ since we will need all of them for the computation of the q-Lagrange polynomial.*

Lemma 35 *$\Lambda_{\mathbf{r},\mathbf{g}}(x)$ can be computed with at most $O_{q^m}(n^2)$ operations.*

Proof Consider again the iterative construction from Proposition 3. At step i we need to compute $\Lambda_i(x) = \Lambda_{i-1}(x) - \frac{\Lambda_{i-1}(g_i)-r_i}{\Pi_{i-1}(g_i)}\Pi_{i-1}(x)$. We need to compute two evaluations of polynomials of q-degree at most i, whose complexity is at most $O_{q^m}(i)$, and a negligible division and difference. Moreover, we need to take the difference of the two polynomials, which is in the order of at most $O_{q^m}(i)$ (because of the degrees). The q-degrees are always at most n, hence for each step we need at most $O_{q^m}(n)$ operations. Since we have n steps, we get the desired complexity order.

Note that computing $\Pi_{\mathbf{g}}(x)$ and $\Lambda_{\mathbf{r},\mathbf{g}}(x)$ ad hoc is much more expensive than using the iterative definition, which is why we used the method from Proposition 3.

Proposition 36 *Algorithm 2 has complexity order $O_{q^m}(n^3)$.*

Proof Once we have the basis of the interpolation module, i.e. after computing $\Lambda_{\mathbf{r},\mathbf{g}}(x)$, the computation of the minimal basis consists of several linearized extended Euclidean algorithms. The complexity order of the linearized EEA is upper bounded by the square of the larger q-degree of the two input polynomials (see e.g. [26, Section 3.1]), thus in our case it is $O_{q^m}(n^2)$. At each step the q-degree decreases in our algorithm, hence there are at most n steps. Overall, the complexity order of Algorithm 2 is upper bounded by $O(n^3)$. Thus, the computation of $\Lambda_{\mathbf{r},\mathbf{g}}(x)$ becomes negligible.

We will now analyze the iterative method of constructing a minimal basis for the interpolation module.

Proposition 37 *Algorithm 3 has complexity order $O_{q^m}(qn^2)$.*

Proof For the iterative computation of the minimal basis from Algorithm 3 we need n steps. In each step we need some polynomial evaluations and differences to compute Δ_i and Γ_i, which needs $O_{q^m}(n)$ operations (similarly to before). Moreover, we need to multiply a linearized polynomial of q-degree at most n by a scalar, which also needs $O_{q^m}(n)$ operations. Similarly, the last step is the composition with $(x^q - g_i^{q-1}x)$, which is analogous to the computation of $\Pi_{\mathbf{g}}(x)$ and is hence in the order of $O_{q^m}(qn)$. Overall we get an upper bound on the complexity of $O(qn^2)$.

As before, if we use a normal basis representation of the extension field, the complexity order of Algorithm 3 can be reduced to $O_{q^m}(n^2)$. If we do so, or if $q \leq n$, the upper bound for the iterative method is better than the one for the algorithm using the extended Euclidean algorithm (EEA). Note though, that our upper bound for the number of iterations needed in Algorithm 2 is very rough and the actual complexity of the algorithm might be much lower. Moreover, one could use the faster version of the Euclidean algorithm from [26] to further decrease the complexity of this algorithm.

We still need to estimate the complexity of the actual decoding algorithm, using our parametrization.

Theorem 38 *Let t be the rank distance between the received word $\mathbf{r}$ and the closest codeword $\mathbf{c} \in \mathbf{C}$. The complexity of Algorithm 1, using Algorithm 3 for the computation of the minimal basis, is upper bounded by*

$$O_{q^m}((q^{m(2t+k-n)} + q)n^2)$$

Thus, the computational complexity of our algorithm is exponential in q if and only if $2t + k - n \geq 0 \iff t > (n-k)/2$, i.e. $\mathbf{r}$ is beyond the unique decoding radius.

Proof The complexity is dominated by the number of different $a(x), b(x)$ we need to consider in Algorithm 1. We know that j runs from 0 to $t - \ell_2 + k - 1$ to find the solutions at distance t. For the largest value of j the parametrization considers all $a(x) \in \mathcal{L}_q(x, q^m)$ with $\mathrm{qdeg}(a(x)) \leq t - \ell_1 + k - 1$ and all monic $b(x) \in \mathcal{L}_q(x, q^m)$ with $\mathrm{qdeg}(b(x)) \leq t - \ell_2 + k - 1$. Hence we get

$$q^{m(t-\ell_1+k)} q^{m(t-\ell_2+k-1)} = q^{m(2t-(\ell_1+\ell_2)+2k-1)} = q^{m(2t+k-n)}$$

possible pairs (where we used Lemma 29 for the last equality). For each such pair $a(x), b(x)$ the symbolic division algorithm for linearized polynomials is executed, which has a complexity order of square of the larger q-degree of the two polynomials (see e.g. [6, 26]). The degrees of our polynomials can be upper bounded by n, hence the symbolic division has order at most $O_{q^m}(n^2)$. Furthermore, we need to consider the complexity of Algorithm 3, hence we get the desired complexity order.

Note that it was already shown in Corollary 32 that only one loop with one symbolic division needs to be executed if $t \leq (n-k)/2$, hence the complexity order in this case is given by the complexities of Algorithm 2 or 3.

In the following we will show what happens beyond the unique decoding radius. For this we first derive another result on ℓ_1.

Lemma 39 *If $t > (n-k)/2$, then $\ell_1 \leq t + k - 1$.*

Proof We will prove this by contradiction. Assume that $\ell_1 > t + k - 1$. It follows right from the degree requirements in Theorem 20 that any linear combination involving $b^{(1)}$ would have higher degree than the one corresponding to distance t. Thus, the element of $\mathfrak{M}(\mathbf{r})$ that corresponds to the closest codeword must be a multiple of $b^{(2)}$. But if a multiple of $b^{(2)}$ fulfills the divisibility requirement, then already $b^{(2)}$ itself needs to fulfill it as well. Hence the message polynomial is the symbolic quotient of $b_1^{(2)}$ by $b_2^{(2)}$ and the algorithm terminates after just computing this quotient. It follows that $\ell_2 = t + k - 1$ and thus, by Lemma 29, $n + k - 1 = \ell_1 + \ell_2 > 2(t + k - 1)$, which is equivalent to $t < (n - k + 1)/2 \iff t \leq (n-k)/2$.

Therefore we can assume $\ell_1 \leq t+k-1$ (and thus $\ell_2 \geq n-t$, by Lemma 29) for the remaining investigation. Hence, we get that

$$n-t \leq \ell_i \leq t+k-1 \quad , \quad i=1,2.$$

If we write $t = \lfloor (n-k)/2 \rfloor + \epsilon$ for some $1 \leq \epsilon \leq n$ we get

$$\left\lceil \frac{n+k}{2} \right\rceil - \epsilon \leq \ell_i \leq \left\lfloor \frac{n+k}{2} \right\rfloor + \epsilon - 1 \quad , \quad i=1,2.$$

We want to show what happens if ϵ is small. Assume that $\epsilon = 1$, then we distinguish two cases.
Case 1: $n+k$ odd
Then the only possible pair is

$$\ell_1 = \ell_2 = \frac{n+k-1}{2} = t+k-1.$$

Hence, the parametrization goes over all $a(x) \in \mathcal{L}_q(x, q^m)$ with $\mathrm{qdeg}(a(x)) \leq 0$ and $b(x) = x$. There are q^m many linear combinations (all scalar multiples of $b^{(1)}$) where the algorithm checks for divisibility.
Case 2: $n+k$ even
Then the possible pairs are

$$\ell_1 = \frac{n+k}{2} - 1 \text{ and } \ell_2 = \frac{n+k}{2} = t+k-1,$$

$$\ell_1 = \frac{n+k}{2} = t+k-1 \text{ and } \ell_2 = \frac{n+k}{2} - 1.$$

In the first case the parametrization goes over all $a(x) \in \mathcal{L}_q(x, q^m)$ with $\mathrm{qdeg}(a(x)) \leq 1$ and $b(x) = x$, which are q^{2m} many possibilities. In the second case the parametrization first only checks $b^{(1)}$ and then goes over all $a(x) \in \mathcal{L}_q(x, q^m)$ with $\mathrm{qdeg}(a(x)) = 0$ and monic $b(x) \in \mathcal{L}_q(x, q^m)$ with $\mathrm{qdeg}(b(x)) = 1$, which are q^{2m} many possibilities in total.

This shows that in the case of one-step ahead decoding, the decoding complexity is still reasonable (especially for small q and m), even though the general complexity is exponential.

5 Conclusion

In this paper we used a parametrization approach for decoding Gabidulin codes with respect to the rank metric. Our main result is that we use this algorithm to compute a list of message polynomials that correspond to *all* codewords that are closest to a given received word.

We developed some results on modules over the ring of linearized polynomials and proved the Predictable Leading Monomial (PLM) property for minimal bases of these modules. Furthermore we defined an interpolation module for a given Gabidulin code and a received word. The decoding algorithm and the parametrization were set inside this interpolation module, using the PLM property as a key ingredient.

To compute a minimal basis of the interpolation module we presented two algorithms – first one using the extended Euclidean algorithm for linearized polynomials and then an iterative algorithm with simple update steps.These algorithms are similar to other known algorithms, but in our setting they can be used for finding all closest codewords which differs from the known algorithms. Moreover, our setting is a novel point of view and we showed that these algorithms actually result in a minimal basis of the interpolation module.

Finally we gave a detailed complexity analysis, showing that our algorithm has polynomial computation complexity if and only if the received word is within the unique decoding radius. Nonetheless we showed that the algorithm is still feasible, if the rank distance between the received word and the code is close to the unique decoding radius, e.g. for one-step-ahead decoding.

References

1. Adams, W.W., Loustaunau, P.: An introduction to Gröbner bases, *Graduate Studies in Mathematics*, vol. 3. American Mathematical Society, Providence, RI (1994)
2. Ali, M., Kuijper, M.: A parametric approach to list decoding of Reed-Solomon codes using interpolation. IEEE Trans. Inform. Theory **57**(10), 6718–6728 (2011). DOI 10.1109/TIT.2011.2165803. URL `http://dx.doi.org/10.1109/TIT.2011.2165803`
3. Delsarte, P.: Bilinear forms over a finite field, with applications to coding theory. Journal of Combinatorial Theory, Series A **25**(3), 226–241 (1978)
4. Gabidulin, E.M.: Theory of codes with maximum rank distance. Problemy Peredachi Informatsii **21**(1), 3–16 (1985)
5. Gabidulin, E.M.: A fast matrix decoding algorithm for rank-error-correcting codes. In: Algebraic coding (Paris, 1991), *Lecture Notes in Comput. Sci.*, vol. 573, pp. 126–133. Springer, Berlin (1992). DOI 10.1007/BFb0034349. URL `http://dx.doi.org/10.1007/BFb0034349`
6. Kötter, R., Kschischang, F.R.: Coding for errors and erasures in random network coding. IEEE Transactions on Information Theory **54**(8), 3579–3591 (2008). DOI 10.1109/TIT.2008.926449
7. Kuijper, M., Schindelar, K.: Minimal Gröbner bases and the predictable leading monomial property. Linear Algebra and its Applications **434**(1), 104–116 (2011)
8. Kuijper, M., Trautmann, A.L.: Gröbner bases for linearized polynomials. In: arXiv:1406.4600 [cs.SC] (2014)
9. Kuijper, M., Trautmann, A.L.: Iterative list-decoding of Gabidulin codes via Gröbner based interpolation. In: arXiv:1405.7152 [cs.IT] (2014)
10. Kuijper, M., Trautmann, A.L.: List decoding Gabidulin codes via interpolation and the Euclidean algorithm. In: arXiv:1404.5716 [cs.IT] (2014)
11. Lidl, R., Niederreiter, H.: Finite Fields. Cambridge University Press, Cambridge, London. Second edition
12. Loidreau, P.: Decoding rank errors beyond the error correcting capability. In: International Workshop on Algebraic and Combinatorial Coding Theory (ACCT), pp. 186–190 (2006)
13. Loidreau, P.: A Welch-Berlekamp like algorithm for decoding Gabidulin codes. In: Coding and cryptography, *Lecture Notes in Comput. Sci.*, vol. 3969, pp. 36–45. Springer, Berlin (2006)
14. Lusina, P., Gabidulin, E., Bossert, M.: Maximum rank distance codes as space-time codes. Information Theory, IEEE Transactions on **49**(10), 2757–2760 (2003)
15. Mahdavifar, H., Vardy, A.: List-decoding of subspace codes and rank-metric codes up to singleton bound. In: Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on, pp. 1488–1492 (2012). DOI 10.1109/ISIT.2012.6283511
16. Ore, O.: On a Special Class of Polynomials. Transactions of the American Mathematical Society **35**, 559–584 (1933)
17. Richter, G., Plass, S.: Fast decoding of rank-codes with rank errors and column erasures. In: Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on, pp. 398–398 (2004). DOI 10.1109/ISIT.2004.1365435

18. Roth, R.M.: Maximum-rank array codes and their application to crisscross error correction. IEEE Transactions on Information Theory **37**(2), 328 –336 (1991). DOI 10.1109/18.75248
19. Sidorenko, V., Bossert, M.: Decoding interleaved gabidulin codes and multisequence linearized shift-register synthesis. In: Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on, pp. 1148–1152 (2010). DOI 10.1109/ISIT.2010.5513676
20. Sidorenko, V., Jiang, L., Bossert, M.: Skew-feedback shift-register synthesis and decoding interleaved Gabidulin codes. IEEE Trans. Inform. Theory **57**(2), 621–632 (2011). DOI 10.1109/TIT.2010.2096032. URL `http://dx.doi.org/10.1109/TIT.2010.2096032`
21. Silberstein, N., Rawat, A.S., Vishwanath, S.: Adversarial error resilience in distributed storage using MRD codes and MDS array codes. arXiv:1202.0800v1 [cs.IT] (2012)
22. Silva, D., Kschischang, F.R.: Fast encoding and decoding of gabidulin codes. In: Information Theory, 2009. ISIT 2009. IEEE International Symposium on, pp. 2858–2862 (2009). DOI 10.1109/ISIT.2009.5205272
23. Silva, D., Kschischang, F.R.: On metrics for error correction in network coding. IEEE Transactions on Information Theory **55**(12), 5479–5490 (2009). DOI 10.1109/TIT.2009.2032817
24. Silva, D., Kschischang, F.R., Kötter, R.: A rank-metric approach to error control in random network coding. IEEE Transactions on Information Theory **54**(9), 3951 –3967 (2008). DOI 10.1109/TIT.2008.928291
25. Trautmann, A.L., Silberstein, N., Rosenthal, J.: List decoding of lifted Gabidulin codes via the Plücker embedding. In: Preproceedings of the International Workshop on Coding and Cryptography (WCC) 2013, pp. 539–549. Bergen, Norway (2013)
26. Wachter-Zeh, A.: Decoding of block and convolutional codes in rank metric. Ph.D. thesis, Ulm University, Germany (2013)
27. Wachter-Zeh, A., Zeh, A.: Interpolation-based decoding of interleaved Gabidulin codes. In: Preproceedings of the International Workshop on Coding and Cryptography (WCC) 2013, pp. 527–537. Bergen, Norway (2013)
28. Xie, H., Yan, Z., Suter, B.: General linearized polynomial interpolation and its applications. In: Network Coding (NetCod), 2011 International Symposium on, pp. 1–4 (2011). DOI 10.1109/ISNETCOD.2011.5978942