Generalized Gabidulin codes over fields of any characteristic

Daniel Augot · Pierre Loidreau · Gwezheneg Robert

Abstract We generalize Gabidulin codes to a large family of fields, non necessarily finite, possibly with characteristic zero. We consider a general field extension and any automorphism in the Galois group of the extension. This setting enables to give several definitions of metrics related to the rank-metric, yet potentially different. We provide sufficient conditions on the given automorphism to ensure that the associated rank metrics are indeed all equal and proper, in coherence with the usual definition from linearized polynomials over finite fields. Under these conditions, we generalize the notion of Gabidulin codes. We also present an algorithm for decoding errors and erasures, whose complexity is given in terms of arithmetic operations. Over infinite fields the notion of code alphabet is essential, and more issues appear that in the finite field case. We first focus on codes over integer rings and study their associated decoding problem. But even if the code alphabet is small, we have to deal with the growth of intermediate values. A classical solution to this problem is to perform the computations modulo a prime ideal. For this, we need study the reduction of generalized Gabidulin codes modulo an ideal. We show that the codes obtained by reduction are the classical Gabidulin codes over finite fields. As a consequence, under some conditions, decoding generalized Gabidulin codes over integer rings can be reduced to decoding Gabidulin codes over a finite field.

Daniel Augot
INRIA Saclay-Île-de-France, and École polytechnique, Palaiseau, France
E-mail: daniel.augot@inria.fr

Pierre Loidreau
DGA MI and IRMAR, Université de Rennes 1
E-mail: pierre.loidreau@univ-rennes1.fr

Gwezheneg Robert
DGA MI
E-mail: gwezheneg.robert@intradef.gouv.fr

1 Introduction

Gabidulin codes and the rank metric were introduced in [Del78] from a combinatorial point of view, and in [Gab85] from an algorithmic and algebraic point of view. They play an important role in coding theory as well as in cryptography. From a coding theory point of view they are well adapted to correct errors and erasures, either occurring along lines (rows or columns) of a matrix, for instance in chip array storage and magnetic tapes [Rot91, BM85], or operating on vector spaces as in the network coding case [KK08]. In the field of cryptography, Gabidulin codes allow to build McEliece-like cryptosystems [GPT91, FL06], with potentially smaller public keys.

Gabidulin codes are typically defined over finite fields using the notion of linearized polynomials. Our goal is to extend the theory of Gabidulin codes to more general type of fields, in particular to number fields. We will particularly focus on the algebraic and algorithmic properties which differ from the finite field case. We make an extensive use of Ore theory of θ-polynomials, which is the natural generalization of linearized polynomials [Ore34, Ore33b].

In **Section 2**, we introduce the ring of θ-polynomials and an evaluation operator which is suitable for our purpose. Then we propose different definitions for the rank metric. We give a framework in which all the definitions are equivalent, generalizing faithfully the finite field case. In our framework, Generalized Gabidulin codes are the codes given by the evaluation of θ-polynomials of bounded degree on a set of points which are linearly independent over the ground field. Similarly to Gabidulin codes, these are also optimal (MRD, maximum rank distance). The reader interested in other applications of θ-polynomials to coding theory and the relations between the different types of evaluations can refer to [BGU07, BU14].

In **Section 3** we study the problem of decoding Generalized Gabidulin codes. Concerning errors, finding an error of rank less than the error-correcting capability can be done by solving a system of linear equations, followed by a left-Euclidean division of θ-polynomials. Concerning errors and erasures, we show that the two known models of erasures (line erasures and network coding erasures) can be deal with by transforming the Generalized Gabidulin code into a Generalized Gabidulin code of smaller length, in which there are no more erasures, only errors.

Therefore, in **Section 4**, we provide an algorithm for decoding errors. Instead of solving directly the above linear system, we derive an algorithm inspired by the so-called Welch-Berlekamp algorithm [BW86, GS92]. This algorithm requires a quadratic number of arithmetic operations, and a very limited amount of storage. We prove that the algorithm decodes up to the error-correcting capability without failures and we give two variants, more involved but which decrease the complexity in practice.

Over infinite fields, the size of the coefficients can grow very rapidly. Therefore, in **Section 5** we address the problem of controlling of the size of coefficients, in the particular case of number fields. Indeed, our basic decoding algorithm has the drawback that the size of intermediate coefficients increases exponentially, doubling at every step. In the case of integer rings, we show that under some conditions, reducing the restricted code modulo a prime ideal gives a convenient classical Gabidulin code over a finite field, which can be use for decoding errors whose magnitude is known (in terms of the size of its coefficients).

Finally we present two detailed examples in **Section 6**. The first one shows a complete run of the decoding algorithm with all the intermediate values. The second

one demonstrates the interest of doing the computations in the residue field, and we provide some timings.

2 Generalization of Gabidulin codes

In this Section, we generalize Gabidulin codes to the case of an arbitrary algebraic extension $K \hookrightarrow L$ of any field K, with an automorphism $\theta \in \mathrm{Aut}_K(L)$. Given θ, we first define θ-polynomials, which are a natural generalization of q-polynomials, described in [Ore33a] and which are classically used to design Gabidulin codes in finite fields. We present properties of θ-polynomials relevant for rank-metric codes, and give proofs that these properties are independent of the underlying field, under an hypothesis $\mathcal{H}_{\dim}$. We give several possible definitions of the *rank metric* and prove that they are equivalent under the hypothesis $L^\theta = K$. Then, the hypotheses $\mathcal{H}_{\dim}$ and $L^\theta = K$ are shown to be equivalent, and hold indeed for cyclic Galois extensions. When these hypotheses are valid, we have a nice framework for studying Gabidulin codes and give their main properties.

2.1 Ore rings and θ-polynomials

A q-polynomial in a finite field extension $\mathbb{F}_q \hookrightarrow \mathbb{F}_{q^m}$ is a polynomial of the form $\sum_i a_i X^{q^i}$, $a_i \in \mathbb{F}_{q^m}$. In coding theory, they are also called linearized polynomial [Ber68, LN97]. When the Frobenius automorphism $x \mapsto x^q$ is replaced by an automorphism $x \mapsto \theta(x)$ of an extension field, these polynomials are called θ-polynomials. Originally they were introduced by Ore in the 1930's (see [Ore33b] for the general theory and [Ore33a] for q-polynomials). In this Section, we recall some useful facts and we give proofs of statements which analog to the finite field case.

Definition 1 (θ-polynomials) Let $K \hookrightarrow L$ be a field extension of finite degree $m = [L : K]$ and let $\theta \in \mathrm{Aut}_K(L)$ be a K-automorphism. A *θ-polynomial* with coefficients in L is an element of the form

$$\sum_{i \geqslant 0} a_i X^i, \ a_i \in L,$$

with a finite number of non-zero a_i's. We denote by $L[X; \theta]$ the *set of θ-polynomials*, provided with the following operations. Let $A(X) = \sum a_i X^i$, $B(X) = \sum b_i X^i \in L[X; \theta]$ and $c \in L$:

- the *addition* is defined component-wise: $A(X) + B(X) = \sum_i (a_i + b_i) X^i$;
- the *(symbolic) product* is defined by $X \cdot c = \theta(c)X$ and $X^i \cdot X^j = X^{i+j}$, and is extended to any $A(X)$ and $B(X)$

$$A(X) \cdot B(X) = \sum_{i,j} a_i \theta^i(b_j) X^{i+j}.$$

Remark 1 This product is called the *symbolic product* (see [Ore33a]) in order to make a distinction from the classical product in polynomial ring over finite fields.

Definition 2 (θ-degree) The *degree* of a θ-polynomial $A(X) = \sum a_i X^i$ is

$$\deg(A) = \begin{cases} -\infty & \text{if } A = 0, \\ \max\{i : a_i \neq 0\} & \text{if } A \neq 0. \end{cases}$$

Then $L[X;\theta]$ is a non commutative ring, whose unit element is 1. It admits no zero divisors and is a left and right Euclidean ring [Ore33b]. A major difference with the polynomial ring $L[X]$ occurs when defining the evaluation of θ-polynomials on scalars. In the usual case, the evaluation of $A(X) \in L[X]$ at $b \in L$, is simply defined by "replace X by b", which is equivalent to computing the remainder of the Euclidean division of $A(X)$ by $X - b$. This equivalence does not hold for θ-polynomials.

Concerning θ-polynomials, there are two different types of evaluations. The first one consists in taking the remainder of the right Euclidean division of $A(X) \in L[X;\theta]$ by the θ-polynomial $X - b$, see [BU14] for instance. The other type is to consider the evaluation through the use of the automorphism θ of the field L. This is the evaluation that we consider in the sequel of the paper.

Definition 3 (Evaluation) Let $A(X) = \sum a_i X^i \in L[X;\theta]$ and $b \in L$. The *(operator) evaluation* of $A(X)$ at b is defined by:

$$A\{b\} = \sum_i a_i \theta^i(b)$$

The properties of this evaluation are as follows.

Proposition 1 *Let $K \hookrightarrow L$, $\theta \in \mathrm{Aut}_K(L)$ and let $A(X)$, $B(X) \in L[X;\theta]$, a, $b \in L$ and $\lambda, \mu \in K$. Then we have*

$$A\{\lambda a + \mu b\} = \lambda A\{a\} + \mu A\{b\},$$
$$(AB)\{a\} = A\{B\{a\}\}.$$

The roots of a θ-polynomial are then naturally defined as follows.

Definition 4 (Roots and root-space) Let $A(X) \in L[X;\theta]$. An element $b \in L$ is a *root* of $A(X)$ if $A\{b\} = 0$. We define the *root-space* of $A(X)$ by

$$\mathrm{Roots}(A(X)) = \{b \in L : A\{b\} = 0\}.$$

From **Proposition 1**, it is obvious that $\mathrm{Roots}(A(X))$ is a K-vector space. For a classical polynomial in $L[X]$, the number of roots is upper bounded by the degree. In the case of θ-polynomials, we are looking for a similar property.

Definition 5 (Hypothesis $\mathcal{H}_{\dim}$) Let $K \hookrightarrow L$ be a field extension and $\theta \in \mathrm{Aut}_K(L)$. We say that $L[X;\theta]$ satisfies the *hypothesis* $\mathcal{H}_{\dim}$ if for all non-zero θ-polynomial $A(X)$ we have
$$\dim_K \mathrm{Roots}(A(X)) \leqslant \deg A(X).$$

We have the following immediate Proposition, which paves the way for the Lagrange interpolation.

Proposition 2 *Suppose that $L[X;\theta]$ has the property $\mathcal{H}_{\dim}$. Let $v_1, \ldots, v_s \in L$ be K-linearly independent, and $A(X) \in L[X;\theta]$ such that $A\{v_i\} = 0, i = 1, \ldots, s$. Then either $A(X) = 0$, or $\deg A(X) \geqslant s$.*

For a given vector space, there exists a θ-polynomial vanishing on the vector space.

Theorem 1 *Let $K \hookrightarrow L$ be a field extension of finite degree m, and $\theta \in \mathrm{Aut}_K(L)$. Let $V \subset L$ be a K-linear subspace of dimension n. There exists a unique monic polynomial $A(X) \in L[X; \theta]$ which vanishes on all $x \in V$. If furthermore $L[X; \theta]$ satisfies $\mathcal{H}_{\dim}$, $A(X)$ has degree exactly n.*

Proof We first prove existence and uniqueness. Since $[L : K]$ is finite, θ has finite order, and we denote s the order of θ, and we have $(X^s - 1)\{x\} = 0$, for all $x \in L$. In particular, $(X^s - 1)\{x\} = 0$, for all $x \in V$, and there exists a non zero polynomial vanishing on V. Let $A(X)$ and $B(X)$ be two non zero monic polynomials both of minimal degree and vanishing on V. They have the same degree and they are both monic. Then, for all $v \in V$, $(A(X) - B(X))\{v\} = 0$, but $A(X) - B(X)$ has lower degree than $A(X)$ or $B(X)$, which contradicts the minimality of the degree. Thus $A(X) = B(X)$.

Assume that $L[X; \theta]$ satisfies $\mathcal{H}_{\dim}$. By **Proposition 2**, the degree of any non-zero θ-polynomial annihilating V is at least n. Now we give an explicit construction of this polynomial: let $(v_1, \cdots, v_s)$ be a basis of V and let the sequence $A_i(X) \in L[X; \theta]$, $i = 0, \ldots, n$, be recursively defined by

$$A_0(X) = X^0, \tag{1}$$

$$A_i(X) = \left(X - \frac{\theta(A_{i-1}\{v_i\})}{A_{i-1}\{v_i\}} \right) \cdot A_{i-1}(X). \tag{2}$$

By induction it is easy to show that, for $i = 1, \ldots, n$, $A_i(X)$ is monic of degree i and that for all $1 \leqslant j \leqslant i$, $A_i\{v_j\} = 0$. This also proves that the fraction in (2) above is well-defined, since $A_{i-1}\{v_i\} \neq 0$ by $\mathcal{H}_{\dim}$. Therefore, $A(X) = A_n(X)$ is monic of degree n and its root-space is V. $\qquad\square$

Definition 6 (Annihilator polynomial) Let $K \hookrightarrow L$ be a field extension, and $\theta \in \mathrm{Aut}_K(L)$. Let $V \subset L$ be a K-linear subspace of dimension n. The *annihilator polynomial* of V is the monic θ-polynomial $\mathcal{A}_V(X) \in L[X; \theta]$ of minimal degree such that for all $v \in V, \mathcal{A}_V\{v\} = 0$. Given a family $\mathbf{g} = (g_1, \ldots, g_n) \in L^n$, we will use the notation $\mathcal{A}_{\mathbf{g}}(X)$ for the annihilator polynomial $\mathcal{A}_V$, where V is the K-linear space spanned by $\mathbf{g}$.

We can also define interpolating polynomials, and when $\mathcal{H}_{\dim}$ holds, interpolating polynomials have the natural expected degree.

Theorem 2 *Let $K \hookrightarrow L$ be a field extension of finite degree $[L : K] = m$ and $\theta \in \mathrm{Aut}_K(L)$ such that $L[X; \theta]$ satisfies $\mathcal{H}_{\dim}$. Let $\mathbf{g} = (g_1, \ldots, g_n)$ be K-linearly independent elements in L and $\mathbf{y} = (y_1, \ldots, y_n) \in L^n$. There is a unique θ-polynomial of degree at most $n - 1$ such that*

$$\mathcal{I}\{g_i\} = y_i, \ 1 \leqslant i \leqslant n.$$

Proof For $i = 1, \ldots, n$, let $\widehat{\mathbf{g}}_i \overset{def}{=} (g_1, \ldots, g_{i-1}, g_{i+1}, \ldots, g_n)$, and consider

$$\mathcal{I}(X) \overset{def}{=} \sum_{i=1}^{n} y_i \frac{\mathcal{A}_{\widehat{\mathbf{g}}_i}(X)}{\mathcal{A}_{\widehat{\mathbf{g}}_i}\{g_i\}} \ .$$

It is easy to check that $\mathcal{I}(X)$ satisfies the conditions of **Theorem 2**. Suppose now that there exists $B(X) \in L[X; \theta]$ of degree $n - 1$ such that $B\{g_i\} = y_i, 1 \leqslant i \leqslant n$. Then $\mathcal{I}(X) - B(X)$ has degree $\leqslant n - 1$ and $(\mathcal{I} - B)\{g_i\} = 0$ for $1 \leqslant i \leqslant n$. The hypothesis $\mathcal{H}_{\dim}$ implies that that $B(X) = \mathcal{I}(X)$ and we have uniqueness. $\square$

Definition 7 (Interpolating polynomial) The polynomial introduced at **Theorem 2** is called the *interpolating θ-polynomial* of $\mathbf{y}$ at $\mathbf{g}$, and is denoted by $\mathcal{I}_{\mathbf{g}, \mathbf{y}}$.

2.2 Rank metrics

Over finite field extensions, the rank metric was studied by Delsarte [Del78] from a combinatorial point of view. Since in this paper our aim is to extend the notion of rank metric to infinite fields, the definitions introduced here are compatible with the finite field case.

Given a field K, we let $\mathcal{M}_{r,c}(K)$ denote the ring of matrices with r rows, c columns and coefficients in K. Given $K \hookrightarrow L$ a field extension, and a matrix M with coefficients in L, we write $\operatorname{rank}_L(M)$ for the maximal number of linearly independent columns over the field L, which is the usual rank of a matrix, and $\operatorname{rank}_K(M)$ for the maximal number of linearly independent columns over the field K.

Definition 8 (Four rank metrics) Let $K \hookrightarrow L$ be a field extension of finite degree $[L : K] = m$, and let $\theta \in \operatorname{Aut}_K(L)$ be an automorphism of order s. Let $\mathcal{B} = (b_1, \ldots, b_m)$ be a K-basis of L, and for $\mathbf{x} = (x_1, \ldots, x_n) \in L^n$, let $x_{ij} \in K$, $i = 1, \ldots, n$, $j = 1, \ldots, m$, be the coordinates of the x_i's in the K-basis $\mathcal{B}$ as follows:

$$x_i = \sum_{j=1}^{m} x_{ij} b_j, \quad i = 1, \ldots, n.$$

Considering

$$\mathbf{B}_{\mathbf{x}, \mathcal{B}} = \begin{pmatrix} x_{1,1} & \cdots & x_{n,1} \\ \vdots & \ddots & \vdots \\ x_{1,m} & \cdots & x_{n,m} \end{pmatrix} \in \mathcal{M}_{m \times n}(K),$$

and

$$\mathbf{V}_{\mathbf{x}, \boldsymbol{\theta}} = \begin{pmatrix} x_1 & \cdots & x_n \\ \vdots & \ddots & \vdots \\ \theta^{s-1}(x_1) & \cdots & \theta^{s-1}(x_n) \end{pmatrix} \in \mathcal{M}_{s \times n}(L),$$

we define the four following *weights* for $\mathbf{x} \in L^n$:

$$w_{\mathcal{A}}(\mathbf{x}) = \deg \mathcal{A}_{\mathbf{x}}(X),$$
$$w_{\theta, L}(\mathbf{x}) = \operatorname{rank}_L \mathbf{V}_{\mathbf{x}, \boldsymbol{\theta}},$$
$$w_{\theta, K}(\mathbf{x}) = \operatorname{rank}_K \mathbf{V}_{\mathbf{x}, \boldsymbol{\theta}},$$
$$w_{\mathcal{B}}(\mathbf{x}) = \operatorname{rank}_K \mathbf{B}_{\mathbf{x}, \mathcal{B}}.$$

Definition 9 (Rank distances) We endow L^n with the distances induced by previous weights:

$$d_{\mathcal{A}}(\mathbf{x}, \mathbf{y}) = w_{\mathcal{A}}(\mathbf{x} - \mathbf{y}),$$
$$d_{\theta, L}(\mathbf{x}, \mathbf{y}) = w_{\theta, L}(\mathbf{x} - \mathbf{y}),$$
$$d_{\theta, K}(\mathbf{x}, \mathbf{y}) = w_{\theta, K}(\mathbf{x} - \mathbf{y}),$$
$$d_{\mathcal{B}}(\mathbf{x}, \mathbf{y}) = w_{\mathcal{B}}(\mathbf{x} - \mathbf{y}).$$

These distances are called *rank distances*.

It is clear that the last three weights induce distance over L^n. We need a proof for $d_{\mathcal{A}}$.

Proposition 3 *The metric $d_{\mathcal{A}}$ is a distance.*

Proof We prove the relevant statements for the associated weight function $w_{\mathcal{A}}$. Let $\mathbf{x} \in L^n$ such that $w_{\mathcal{A}}(\mathbf{x}) = 0$. This means that the annihilator polynomial of $\mathbf{x}$ has degree 0, and, being monic, it is the polynomial $A(X) = 1 \in L[X, \theta]$. Now, for $i = 1, \ldots, n$, we have $A\{x_i\} = 1\{x_i\} = x_i = 0$. Thus $w_{\mathcal{A}}(\mathbf{x}) = 0$ implies $\mathbf{x} = 0$. Now we have easily that $w_{\mathcal{A}}(-\mathbf{x}) = w(\mathbf{x})$, since, for any $A(X) \in L[X; \theta]$ which vanishes on $\mathbf{x}$, we have $A\{-x_i\} = -A\{x_i\} = 0$, $i = 1, \ldots, n$, and the annihilator polynomials of $\mathbf{x}$ and $-\mathbf{x}$ are the same.

We finally have to prove that, given $\mathbf{x}_1, \mathbf{x}_2 \in L^n$, $w_{\mathcal{A}}(\mathbf{x}_1 + \mathbf{x}_2) \leqslant w_{\mathcal{A}}(\mathbf{x}_1) + w_{\mathcal{A}}(\mathbf{x}_2)$. Let $w_{\mathcal{A}}(\mathbf{x}_1) = w_1$ and $w_{\mathcal{A}}(\mathbf{x}_2) = w_2$, and consider the following elements in L^n, where addition and θ are applied component-wise:

$$\mathbf{y}_0 = \mathbf{x}_1 + \mathbf{x}_2,$$
$$\mathbf{y}_1 = \theta(\mathbf{x}_1 + \mathbf{x}_2) = \theta(\mathbf{x}_1) + \theta(\mathbf{x}_2),$$
$$\vdots$$
$$\mathbf{y}_{w_1 + w_2} = \theta^{w_1 + w_2}(\mathbf{x}_1 + \mathbf{x}_2) = \theta(\mathbf{x}_1)^{w_1 + w_2} + \theta(\mathbf{x}_2)^{w_1 + w_2}.$$

Now using that $\theta^i(\mathbf{x}_1 + \mathbf{x}_2) = \theta^i(\mathbf{x}_1) + \theta^i(\mathbf{x}_2)$, $i = 1 \ldots, w_1 + w_2$, and considering the rest of the right division of X^i by $\mathcal{A}_{\mathbf{x}_1}(X)$ and by $\mathcal{A}_{\mathbf{x}_2}(X)$, we see that $\theta^i(\mathbf{x}_1) + \theta^i(\mathbf{x}_2)$ belong the L-vector space

$$V_{\mathbf{x}_1, \mathbf{x}_2} = \mathrm{Vect}_L \left\langle \mathbf{x}_1, \theta(\mathbf{x}_1), \ldots, \theta(\mathbf{x}_1)^{w_1 - 1}, \mathbf{x}_2, \theta(\mathbf{x}_2), \ldots, \theta^{w_2 - 1}(\mathbf{x}_2) \right\rangle.$$

Now, since $\dim_L V_{\mathbf{x}_1, \mathbf{x}_2} \leqslant w_1 + w_2$, there exists an L-linear non zero dependency between the $w_1 + w_2 + 1$ vectors $\mathbf{y}_0, \ldots, \mathbf{y}_{w_1 + w_2}$. Then, by **Proposition 2**, the annihilator polynomial of $\mathbf{x}_1 + \mathbf{x}_2$ has degree less than or equal to $w_1 + w_2$. □

The natural question about the relationship between the different induced metrics is answered in the following Theorems.

Theorem 3 *For all $\mathbf{x} \in L^n$, $w_{\mathcal{A}}(\mathbf{x}) = w_{\theta, L}(\mathbf{x})$.*

Proof The case of $\mathbf{x} = 0$ being trivial, let us consider $\mathbf{x} = (x_1, \ldots, x_n) \neq 0 \in L^n$. We first show that $w_{\theta, L}(\mathbf{x}) \leqslant w_{\mathcal{A}}(\mathbf{x})$. Let $w = w_{\mathcal{A}}(\mathbf{x})$. Let $\mathcal{A}_{\mathbf{x}}(X) = \sum_{i=0}^{w} a_i X^i$ be the monic annihilator polynomial of $\mathbf{x}$. Then, let L_i, $i = 0, \ldots, s-1$ be the $i+1$-th row of $\mathbf{V}_{\mathbf{x}, \theta}$. We have $L_w = -\sum_{i=0}^{w-1} a_{i-1} L_i$ and the $(w+1)$-th row is a linear combination of the previous ones. Then, applying recursively θ on this relation, we can express any row of index larger than or equal to $w + 1$ as a linear combination of the w first rows.

Thus, $\mathbf{V}_{\mathbf{x},\theta}$ has at most $w_{\mathcal{A}}(\mathbf{x})$ linearly L-independent rows which is the same as the number of L-independent columns. Thus $w_{\theta,L}(\mathbf{x}) \leqslant w_{\mathcal{A}}(\mathbf{x})$.

Conversely, let $w = w_{\theta,L}(\mathbf{x})$. Let $1 \leqslant u \leqslant w$ be the smallest index such that $L_1, \ldots, L_u$ are K-Linearly independent. Since, by construction of $\mathbf{V}_{\mathbf{x},\theta}$, $L_{i+1} = \theta(L_i)$ where θ acts on all the components of the row L_i, any row L_i, $i \geqslant u+1$, is a linear combination of the rows $L_1, \ldots, L_u$. Therefore, $u = w$, and there is a linear combination such that $L_w + \sum_{i=0}^{w-1} \mu_i L_i = 0$, and the polynomial $X^w + \sum_{i=0}^{w-1} \mu_i X^i$ annihilates $\mathbf{x}$. Hence $w_{\mathcal{A}}(\mathbf{x}) \leqslant w = w_{\theta,L}(\mathbf{x})$.   □

For the next inequalities, we have to compare the number of linearly independent columns of two matrices. For that, we will consider a linear combination of columns of the first one, and show that it gives a linear combination in the second matrix. Since the order of the columns, reflecting the ordering of the x_i's in $\mathbf{x}$, does not matter, w.l.o.g. we can order them such that the r first columns are linearly independent, where r denotes the rank of the matrix.

Theorem 4 *For all* $\mathbf{x} \in L^n$, $w_{\theta,K}(\mathbf{x}) = w_{\mathcal{B}}(\mathbf{x})$.

Proof The case of 0 being trivial, let us consider $\mathbf{x} = (x_1, \ldots, x_n) \neq 0 \in L^n$. We show that $w_{\theta,K}(\mathbf{x}) \leqslant w_{\mathcal{B}}(\mathbf{x})$. Let $w = w_{\mathcal{B}}(\mathbf{x}) = \mathrm{rank}_K(\mathbf{B}_{\mathbf{x},\mathcal{B}})$, and w.l.o.g. assume that the first w columns of $\mathbf{B}_{\mathbf{x},\mathcal{B}}$ are linearly independent. Consider the u-th column expressed as a linear combination of the first w columns of $\mathbf{B}_{\mathbf{x},\mathcal{B}}$:

$$\begin{pmatrix} x_{u,1} \\ \vdots \\ x_{u,m} \end{pmatrix} = \lambda_1 \begin{pmatrix} x_{1,1} \\ \vdots \\ x_{1,m} \end{pmatrix} + \cdots + \lambda_w \begin{pmatrix} x_{w,1} \\ \vdots \\ x_{w,m} \end{pmatrix},$$

where $\lambda_i \in K$ for $i = 1, \ldots, w$. For $i = 1, \ldots, n$, writing

$$x_i = \sum_{j=1}^{m} x_{ij} b_j,$$

we have

$$x_u = \lambda_1 x_1 + \cdots + \lambda_w x_w.$$

By applying the K-automorphism θ^k, for $k = 1, \ldots, s$, we get

$$\theta^k(x_u) = \lambda_1 \theta^k(x_1) + \cdots + \lambda_w \theta^k(x_w).$$

Hence,

$$\begin{pmatrix} x_u \\ \vdots \\ \theta^{s-1}(x_u) \end{pmatrix} = \lambda_1 \begin{pmatrix} x_1 \\ \vdots \\ \theta^{s-1}(x_1) \end{pmatrix} + \cdots + \lambda_w \begin{pmatrix} x_w \\ \vdots \\ \theta^{s-1}(x_w) \end{pmatrix},$$

and a linear combination in $\mathbf{B}_{\mathbf{x},\mathcal{B}}$ implies a linear combination in $\mathbf{V}_{\mathbf{x},\theta}$, thus the K-rank of $\mathbf{V}_{\mathbf{x},\theta}$ is at most the K-rank of $\mathbf{B}_{\mathbf{x},\mathcal{B}}$.

Now, we show that $w_{\mathcal{B}}(\mathbf{x}) \leqslant w_{\theta,K}(x)$. Let $w = w_{\theta,K}(\mathbf{x}) = \mathrm{rank}_K(\mathbf{V}_{\mathbf{x},\theta})$. Consider the expression of the u-th column as a K-linear combination of the w first columns of $\mathbf{V}_{\mathbf{x},\theta}$:

$$\begin{pmatrix} x_u \\ \vdots \\ \theta^{s-1}(x_u) \end{pmatrix} = \lambda_1 \begin{pmatrix} x_1 \\ \vdots \\ \theta^{s-1}(x_1) \end{pmatrix} + \cdots + \lambda_w \begin{pmatrix} x_w \\ \vdots \\ \theta^{s-1}(x_w) \end{pmatrix},$$

where $\lambda_i \in K$, for all $i = 1, \ldots, w$. Then, by considering the first row, we have $x_u = \lambda_1 x_1 + \cdots + \lambda_w x_w$, which can be expanded in the basis $\mathcal{B}$ as follows:

$$x_{u,j} = \sum_{i=1}^{w} \lambda_i x_{i,j}, 1 \leqslant j \leqslant m.$$

Thus,

$$\begin{pmatrix} x_{u,1} \\ \vdots \\ x_{u,m} \end{pmatrix} = \lambda_1 \begin{pmatrix} x_{1,1} \\ \vdots \\ x_{1,m} \end{pmatrix} + \cdots + \lambda_w \begin{pmatrix} x_{w,1} \\ \vdots \\ x_{w,m} \end{pmatrix}$$

so $w_{\mathcal{B}}(\mathbf{x}) = \operatorname{rank}_K(X_{\mathcal{B}}) \leqslant w$. $\qquad\square$

Proposition 4 *We have the following inequality, for all $\mathbf{x} \in L^n$:*

$$w_{\mathcal{A}}(\mathbf{x}) = w_{\theta,L}(\mathbf{x}) \leqslant w_{\theta,K}(\mathbf{x}) = w_{\mathcal{B}}(\mathbf{x}).$$

Proof A linear combination with coefficients in K is also a linear combination with coefficients in L, so $w_{\theta,L}(\mathbf{x}) \leqslant w_{\theta,K}(\mathbf{x})$.

$\qquad\square$

Now we introduce a condition for all these weights to be equal.

Proposition 5 *If $L^\theta = K$, where $L^\theta = \{x \in L : \theta(x) = x\}$ is the fixed field of θ, then, for all $\mathbf{x} \in L^n$,*

$$w_{\mathcal{A}}(\mathbf{x}) = w_{\theta,L}(\mathbf{x}) = w_{\theta,K}(\mathbf{x}) = w_{\mathcal{B}}(\mathbf{x}).$$

Proof It suffices to show that for all $\mathbf{x} = (x_1, \ldots, x_n) \in L^n$, $w_{\theta,K}(\mathbf{x}) \leqslant w_{\theta,L}(\mathbf{x})$. Let $w = w_{\theta,L}(\mathbf{x}) = \operatorname{rank}_L(\mathbf{V}_{\mathbf{x},\boldsymbol{\theta}})$. Consider the expression of the u-th column as an L-linear combination of of the w first columns of $\mathbf{V}_{\mathbf{x},\boldsymbol{\theta}}$ which form a basis of the column space of $\mathbf{V}_{\mathbf{x},\boldsymbol{\theta}}$:

$$\begin{pmatrix} x_n \\ \vdots \\ \theta^{s-1}(x_n) \end{pmatrix} = \mu_1 \begin{pmatrix} x_1 \\ \vdots \\ \theta^{s-1}(x_1) \end{pmatrix} + \cdots + \mu_w \begin{pmatrix} x_w \\ \vdots \\ \theta^{s-1}(x_w) \end{pmatrix}, \qquad (3)$$

with $\mu_i \in L$, $i = 1, \ldots, w$. By applying θ, we obtain

$$\begin{pmatrix} \theta^1(x_u) \\ \vdots \\ \theta^s(x_u) \end{pmatrix} = \theta(\mu_1) \begin{pmatrix} \theta^1(x_1) \\ \vdots \\ \theta^s(x_1) \end{pmatrix} + \cdots + \theta(\mu_w) \begin{pmatrix} \theta^1(x_w) \\ \vdots \\ \theta^s(x_w) \end{pmatrix}.$$

Since s is the order of θ, i.e. $\theta^s = Id$, by reordering the lines, we also have

$$\begin{pmatrix} x_n \\ \vdots \\ \theta^{s-1}(x_n) \end{pmatrix} = \theta(\mu_1) \begin{pmatrix} x_1 \\ \vdots \\ \theta^{s-1}(x_1) \end{pmatrix} + \cdots + \theta(\mu_w) \begin{pmatrix} x_w \\ \vdots \\ \theta^{s-1}(x_w) \end{pmatrix}.$$

Since the first w columns form a basis of the column space, the decomposition (3) is unique. Therefore, $\mu_i = \theta(\mu_i)$, i.e. $\mu_i \in L^\theta = K$, for $i = 1, \ldots, n$. This implies that $w_{\theta,K}(\mathbf{x}) = \operatorname{rank}_K(\mathbf{V}_{\mathbf{x},\boldsymbol{\theta}}) \leqslant w_{\theta,L}(\mathbf{x}) = \operatorname{rank}_L(\mathbf{V}_{\mathbf{x},\boldsymbol{\theta}})$. $\qquad\square$

When $L^\theta = K$, the four metrics previously defined are identical. This metric is called *the* rank metric, and denoted by $w(\mathbf{x})$.

2.3 A framework for studying Gabidulin codes

From previous Sections, we have seen that when $L[X; \theta]$ satisfies $\mathcal{H}_{\dim}$, the dimension of the root-space of a θ-polynomial $A(X)$ is at most equal to the degree of $A(X)$. Also, when L^{θ}, the field fixed by θ, is K, we have that all the defined metrics are equal. We first present two examples, of a bad behaved and of well behaved extensions.

Example 1 Let $K = \mathbb{Q}[j] = \mathbb{Q}[X]/(X^2 + X + 1)$, which is a field, which contains the sixth roots of the unity $\{1, j, j^2, -1, -j, -j^2\}$ where $j^3 = 1$. We build an extension L of K by adding a sixth root of 2, denoted by α, and $\alpha^6 = 2$. Since square and cubic roots of 2 are not in K, we get the following *Kummer extension* [Neu99, §IV.3]:

$$L = K[\alpha] = K[Y]/(Y^6 - 2).$$

Hence $[L : K] = 6$ and any K-automorphism of L is uniquely defined by the image of α, which must be a root of $Y^6 - 2$. Let us consider $\theta_1 : \alpha \mapsto j\alpha$, which has order 3: $\theta_1^3 = Id_L$. Then $\mathcal{H}_{\dim}$ is not verified, as can be seen by considering the θ_1-polynomial $A = X - 1$. One can check that $\mathrm{Roots}(X - 1)$ contains 1 and α^3, therefore has K-dimension 2, which is twice the degree of A. We also have that the fixed field L^{θ_1} is spanned by 1 and α^3. Therefore $K \neq L^{\theta_1}$. The rank metrics are clearly different:

$$w_{\mathcal{A}}(1, \alpha, \alpha^3, \alpha^4) = 2 \ < \ w_{\mathcal{B}}(1, \alpha, \alpha^3, \alpha^4) = 4.$$

Indeed, we have that $j - (j + 1)X + X^2$ vanishes on $(1, \alpha, \alpha^3, \alpha^4)$, and thus

$$w_{\mathcal{A}}(1, \alpha, \alpha^3, \alpha^4) = \deg\left(j - (j + 1)X + X^2\right) = 2;$$

while

$$w_{\mathcal{B}}(1, \alpha, \alpha^3, \alpha^4) = \mathrm{rank}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} = 4.$$

Example 2 We consider the field extension of Example 1, and now consider $\theta_2 : \alpha \mapsto (j+1)\alpha$. The automorphism θ_2 has order 6. We have that $\mathrm{Roots}(X - 1)$ has dimension 1 and $L^{\theta_2} = K$, which implies the equality of the metrics, for example:

$$w_0(1, \alpha, \alpha^3, \alpha^4) = w_3(1, \alpha, \alpha^3, \alpha^4) = 4.$$

Indeed, we have:

$$w_{\mathcal{A}}(1, \alpha, \alpha^3, \alpha^4) = \deg\left(j - (j + 1)X^2 + X^4\right) = 4;$$

and

$$w_{\mathcal{B}}(1, \alpha, \alpha^3, \alpha^4) = \mathrm{rank}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} = 4.$$

In this case, $\mathcal{H}_{\dim}$ is verified.

These examples show that there is a connection between the hypothesis $\mathcal{H}_{\dim}$ and $L^{\theta} = K$. The following Theorem establishes this link.

Theorem 5 *Let $K \hookrightarrow L$ be a field extension of finite degree $[L : K] = m$, and let $\theta \in \mathrm{Aut}_K(L)$. The following statements are equivalent:*

- *i the subfield L^{θ} is K,*
- *ii the weights $w_{\mathcal{A}}$ and $w_{\mathcal{B}}$ are equal,*
- *iii the annihilator polynomial of n K-linearly independent elements of L has degree exactly n,*
- *iv the dimension of the root-space of any non-zero polynomial is upper-bounded by its degree.*

Proof

$i \Rightarrow ii$ This implication has been proved in **Proposition 5**.

$ii \Rightarrow iii$ Let $\mathbf{v} = (v_1, \dots, v_n)$ be a vector of K-linearly elements of K, meaning that $w_{\mathcal{B}}(\mathbf{v}) = n$. The equivalence is a direct consequence of the definition of $w_{\mathcal{A}}$.

$iii \Rightarrow iv$ Let $P(X)$ be a polynomial of degree n and $(v_1, \dots, v_u)$ be a basis of its roots space. Let $\mathcal{A}(X)$ denotes the annihilator polynomial of the v_i's. By iii, its degree is u. We compute the Euclidean division: $P(X) = Q(X)\mathcal{A}(X) + R(X)$, where $\deg R(X) < \deg \mathcal{A}(X)$. We get that $R\{v_i\} = 0$, but since the annihilator has minimal degree, we deduce that $R(X) = 0$. Thus, $P(X)$ is a multiple of $\mathcal{A}(X)$, and $u \leqslant n$.

$iv \Rightarrow i$ L^{θ} is the roots space of $X - 1$, applying $\mathcal{H}_{\dim}$ to this polynomial gives that L^{θ} has dimension 1 and contains K, so it is K. $\square$

Since θ has a finite order, Artin's Lemma establishes that $L^{\theta} \hookrightarrow L$ is a Galois extension of Galois group $\langle \theta \rangle$. Therefore, if the condition i. above $(K = L^{\theta})$ is satisfied, then $K \hookrightarrow L$ is a Galois extension with Galois group $\mathrm{Aut}_K(L) = \langle \theta \rangle$. Thus, a proper framework to design codes in rank metric and the generalization of Gabidulin codes is to consider a cyclic Galois extension $K \hookrightarrow L$.

Finite fields extensions provided with the Frobenius automorphism are examples of such extensions. Concerning number fields, cyclotomic and Kummer extensions are simple cyclic Galois extension, with explicit generators of the Galois group. Similarly, concerning function fields, Kummer and Artin-Schreier extensions also provide cyclic Galois extensions which are easy to deal with.

2.4 Generalized Gabidulin codes

Gabidulin codes were defined in 1985 by Gabidulin for finite fields [Gab85]. They consist in the evaluation of a q-polynomial of bounded degree k at n values. They are the analogue of Reed-Solomon codes in the rank metric. In this part, we give the Singleton bound for codes in the rank metric, and we generalized the construction of Gabidulin codes. Then, we give basic properties of these codes, with proofs that does not rely on the finiteness of the field.

We consider $K \hookrightarrow L$ a cyclic Galois extension of degree $[L : K] = m$ and $\langle \theta \rangle = \mathrm{Aut}_K(L)$. From previous considerations, we endow the vector space L^n with *the rank metric w*, using any of the four equivalent definitions.

Definition 10 Under the above setting, a $[n,k,d]_r$ CODE $\mathcal{C}$ is an L-linear subspace of dimension k of the vector space L^n with minimum rank distance d, *i.e.* $d = \min_{\mathbf{x} \in \mathcal{C} \setminus 0} w(\mathbf{x})$.

With these parameters, we have

Proposition 6 (Singleton Bound) *Let $\mathcal{C}$ be a $[n,k,d]_r$-code over L. Then $d \leqslant n - k + 1$.*

Proof Consider the projection map π on the first $n - d + 1$ coordinates:

$$
\pi : \qquad \begin{aligned} \mathcal{C} &\to L^{n-d+1} \\ (x_1, \ldots, x_n) &\mapsto (x_1, \ldots, x_{n-d+1}) \end{aligned}
$$

and let $\mathbf{x} \in \mathcal{C}$ be such that $\pi(\mathbf{x}) = 0$, i.e. $\mathbf{x} = (0, \ldots, 0, x_{n-d+2}, \ldots, x_n)$. The number of non zero coordinates of $\mathbf{x}$ is less than or equal to $n - (n - d + 2) + 1 = d - 1$. Then $\mathcal{A}_{\mathbf{x}}(X)$, the annihilator polynomial of x, right divides the left lowest common multiple of

$$
X - \frac{\theta(x_{n-d+2})}{x_{n-d+2}}, \ldots, X - \frac{\theta(x_n)}{x_n}
$$

which has degree less than or equal to $d - 1$. Thus $w(\mathbf{x}) \leqslant d - 1$: since the minimum distance of $\mathcal{C}$ is d, $x = 0$, and π is injective. As a consequence $k \leqslant n - d + 1$. $\qquad\square$

Definition 11 (MRD codes) When $d = n - k + 1$, the code $\mathcal{C}$ is a *MRD (Maximum Rank Distance) code.*

Now we present the generalization of Gabidulin codes.

Definition 12 (Generalized Gabidulin code) Let $K \hookrightarrow L$ be a cyclic Galois extension of degree $[L : K] = m$ and $\mathrm{Aut}_K(L) = \langle \theta \rangle$. Let $k \leqslant n \leqslant m$ be integers and $\mathbf{g} = (g_1, \ldots, g_n)$ be a vector of K-linearly independent elements of L. The vector $\mathbf{g}$ is called the *support* of the code. The *generalized Gabidulin code* $\mathrm{Gab}_{\theta,k}(\mathbf{g})$ is

$$
\mathrm{Gab}_{\theta,k}(\mathbf{g}) = \{(f\{g_1\}, \ldots, f\{g_n\}) \; : \; f(X) \in L[X;\theta], \deg f(X) < k\}. \tag{4}
$$

Remark 2 The case $k = n$ is useful for the theory. This case will appear for decoding algorithms when there is the maximal number of erasures.

A codeword $\mathbf{c} = (c_1, \ldots, c_n)$ of $\mathrm{Gab}_{\theta,k}(\mathbf{g})$ has coefficient in L^n. The generalized Gabidulin code $\mathrm{Gab}_{\theta,k}(\mathbf{g})$ has the following generating matrix

$$
\begin{pmatrix} \theta^0(g_1) & \cdots & \theta^0(g_n) \\ \vdots & \ddots & \vdots \\ \theta^{k-1}(g_1) & \cdots & \theta^{k-1}(g_n) \end{pmatrix}. \tag{5}
$$

Using a K-basis $\mathcal{B} = (b_1, \ldots, b_m)$ of L, writing $c_j = \sum_{i=1}^m c_{ij} b_j$, $c_{ij} \in K$, $i = 1 \ldots m$, $j = 1 \ldots, n$, the codeword $\mathbf{c}$ can alternatively be written in matrix form:

$$
\mathbf{c} = \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{m1} & \cdots & c_{mn} \end{pmatrix} \in \mathcal{M}_{m \times n}(K).
$$

Example 3 We consider the cyclotomic extension $\mathbb{Q} \hookrightarrow \mathbb{Q}[X]/(1+X+\cdots+X^6) = \mathbb{Q}[\alpha]$, where $1 + \alpha + \cdots + \alpha^6 = 0$, and the automorphism defined by $\theta : \alpha \mapsto \alpha^3$. In this setting, $\mathbb{Q} = K$ and $\mathbb{Q}[\alpha] = L$. Let $\mathcal{B} = (1, \alpha, \ldots, \alpha^5)$ be the considered K-basis of L. Let the support be $\mathbf{g} = (1, \alpha, \ldots, \alpha^5)$, and we build the corresponding generalized Gabidulin code of length $n = 6$, dimension $k = 2$ and minimum distance $d = 5$.

To encode the θ-polynomial $f = \alpha^2 + \alpha^5 X$, we compute the evaluations of f on the support:

$$
\begin{aligned}
f\{1\} &= \alpha^2 + \alpha^5, \\
f\{\alpha\} &= \alpha + \alpha^3, \\
f\{\alpha^2\} &= 2\alpha^4, \\
f\{\alpha^3\} &= \alpha^5 + 1 \\
f\{\alpha^4\} &= \alpha^3 + \alpha^6 \\
&= -1 - \alpha - \alpha^2 - \alpha^4 - \alpha^5 \\
f\{\alpha^5\} &= \alpha^6 + 1 \\
&= -\alpha - \alpha^2 - \alpha^3 - \alpha^4 - \alpha^5
\end{aligned}
$$

Then, the corresponding codeword is the following vector:

$$
\mathbf{c}(f) = \left(f\{\alpha\}, \ldots, f\{\alpha^5\} \right) \in \mathbb{Q}[\alpha]^6;
$$

or the following matrix obtain by expanding the components of the vector over the basis $\mathcal{B}$:

$$
\mathbf{C}(f) = \begin{pmatrix}
0 & 0 & 0 & 1 & -1 & 0 \\
0 & 1 & 0 & 0 & -1 & -1 \\
1 & 0 & 0 & 0 & -1 & -1 \\
0 & 1 & 0 & 0 & 0 & -1 \\
0 & 0 & 2 & 0 & -1 & -1 \\
1 & 0 & 0 & 1 & -1 & -1
\end{pmatrix} \in \mathcal{M}_{6,6}(\mathbb{Q}).
$$

In this example, the above matrix will have a large enough rank.

We formalize this property in the following Proposition.

Proposition 7 *Under the conditions of* **Definition 12**, $\mathrm{Gab}_{\theta,k}(\mathbf{g})$ *is an MRD-code.*

Proof Let $\mathbf{c} = (c_1, \ldots, c_n)$ be a non-zero codeword of rank weight w. From **Theorem 1**, there is a non-zero θ-polynomial $U(X)$ of degree w which vanishes on all the c_i's. There also exists a non-zero θ-polynomial $f(X)$ of degree $\leqslant k-1$ such that $c_i = f\{g_i\}$, for all $i = 1, \ldots, n$. We have

$$
U(X) \cdot f(X)\{g_i\} = U\{c_i\} = 0, 1 \leqslant i \leqslant n.
$$

Thus $U(X) \cdot f(X)$ is a non-zero θ-polynomial of degree $\leqslant w + k - 1$ which vanishes on the K-vector space of dimension n generated by the g_i's. So we have $n \leqslant w + k - 1$, i.e. $w \geqslant n - k + 1$. Therefore $n - k + 1 \leqslant d$. From Singleton bound we have $d \leqslant n - k + 1$: $d = n - k + 1$, and the code is MRD. $\qquad\square$

Given the usual scalar product, we recall that the dual, or orthogonal, of a code is the set of vectors orthogonal to all the codewords.

Proposition 8 *A parity-check matrix of* $\mathrm{Gab}_{\theta,k}(\mathbf{g})$ *has the form*

$$\mathbf{H} = \begin{pmatrix} \theta^0(h_1) & \cdots & \theta^0(h_n) \\ \vdots & \ddots & \vdots \\ \theta^{k-1}(h_1) & \cdots & \theta^{k-1}(h_n) \end{pmatrix}, \tag{6}$$

for some $h_1, \ldots, h_n \in L^n$ *which are* K-*linearly independent and satisfy*

$$\begin{pmatrix} \theta^{-d+2}(g_1) & \cdots & \theta^{-d+2}(g_n) \\ \vdots & \ddots & \vdots \\ \theta^{n-d}(g_1) & \cdots & \theta^{n-d}(g_n) \end{pmatrix} \begin{pmatrix} h_1 \\ \vdots \\ h_n \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Moreover the vector $(h_1, \ldots, h_n) \in L^n$ *is unique up to a multiplicative factor in* L.

Proof The equation $\mathbf{G} \cdot \mathbf{H}^T = 0$, where $\mathbf{G}$ denotes the generating matrix of the code (5) is equivalent to

$$\sum_{i=1}^n \theta^a(g_i)\theta^b(h_i) = 0, \quad 0 \leqslant a \leqslant k-1, \quad 0 \leqslant b \leqslant d-2.$$

By applying, for all $b = 0, \ldots, k-1$, the automorphism θ^{-b} we obtain

$$\sum_{i=1}^n \theta^c(g_i)h_i = 0, \quad -n+k+1 \leqslant c \leqslant k-1,$$

which gives the system

$$\begin{pmatrix} \theta^0(\tilde{g}_1) & \cdots & \theta^0(\tilde{g}_n) \\ \vdots & \ddots & \vdots \\ \theta^{n-2}(\tilde{g}_1) & \cdots & \theta^{n-2}(\tilde{g}_n) \end{pmatrix} \begin{pmatrix} h_1 \\ \vdots \\ h_n \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{7}$$

with $\tilde{g}_i = \theta^{-(d-2)}(g_i)$. Since the g_i's are K-linearly independent, so are the $\tilde{g}_i$'s. Therefore, $w_{\theta,K}(\tilde{\mathbf{g}})$, the K-rank weight of $\tilde{\mathbf{g}} = (\tilde{g}_1, \ldots, \tilde{g}_n)$, is equal to n. From **Theorem 5** the L-rank of the matrix

$$\begin{pmatrix} \theta^0(\tilde{g}_1) & \cdots & \theta^0(\tilde{g}_n) \\ \vdots & \ddots & \vdots \\ \theta^{m-1}(\tilde{g}_1) & \cdots & \theta^{m-1}(\tilde{g}_n) \end{pmatrix}$$

is also n. Thus, the kernel of equation (7) has L-dimension 1. Let $(h_1, \ldots, h_n) \neq 0$ belong to the kernel. Now, we prove that the h_i's are K-linearly independent. Consider a linear dependency

$$\sum_{i=1}^n \lambda_i h_i = 0, \quad \lambda_i \in K.$$

If the matrix, that we call A, consisting in (7) augmented with the row $(\lambda_1, \ldots, \lambda_n)$, as a last row, has rank n, then the h_i's are all zero. Since this is not the case, A has rank at most $n-1$ and there exists $\mu_0, \ldots, \mu_{n-2} \in L^{n-1}$ such that

$$\lambda_i = \mu_0 \theta^0(\tilde{g}_i) + \cdots + \mu_{n-2}\theta^{n-2}(\tilde{g}_i), \quad i = 1, \ldots, n,$$

since matrix in (7) has rank $n - 1$. Let $M = \sum_{i=0}^{n-2} \mu_i X^i \in L[X; \theta]$. Since for all i, $\lambda_i \in K$, the λ_i's are also roots of $X - 1$. Therefore, from the properties of the evaluation of θ-polynomials in **Proposition 1**, we have, for $i = 1, \ldots, n$:

$$(X - 1) \cdot M \{\tilde{g}_i\} = (X - 1) \{M \{\tilde{g}_i\}\} = (X - 1) \{\lambda_i\} = 0.$$

The θ-polynomial $(X - 1) \cdot M$ has degree $n - 1$ with n K-linearly independent roots, by **Proposition 2** it is the zero polynomial. This implies that all the μ_i are zero, and for $i = 1, \ldots, n$, $\lambda_i = 0$, i.e. the h_i's are K-linearly independent. $\qquad \square$

Corollary 1 *The dual of a generalized Gabidulin code is a generalized Gabidulin code.*

3 Decoding Generalized Gabidulin Codes

Let $K \hookrightarrow L$ be a cyclic Galois extension of finite degree $m = [L : K]$ and θ a K-automorphism such that $\mathrm{Aut}_K(L) = \langle \theta \rangle$. We consider a generalized Gabidulin code $\mathrm{Gab}_{\theta,k}(\mathbf{g})$ with parameters $[n, k, d]_r$, for some support $\mathbf{g} = (g_1, \ldots, g_n) \in L^n$, such that the g_i's are K-linearly independent. The problem of decoding the code $\mathrm{Gab}_{\theta,k}(\mathbf{g})$ is classically stated as follows. Let $\mathbf{y} \in L^n$ be such that

$$\mathbf{y} = \mathbf{c}(f) + \mathbf{e} + \boldsymbol{\varepsilon} = \mathbf{c} + \mathbf{e} + \boldsymbol{\varepsilon},$$

where $\mathbf{c} = \mathbf{c}(f) = f\{\mathbf{g}\} \in \mathrm{Gab}_{\theta,k}(\mathbf{g})$ is the codeword corresponding to the evaluation of $f(X) \in L[X; \theta]$, $\mathbf{e}$ and $\boldsymbol{\varepsilon}$ are respectively the vector of errors and erasures, which are "small". The goal is to recover $\mathbf{c} \in \mathrm{Gab}_{\theta,k}(\mathbf{g})$, or, equivalently, $f(X) \in L[X; \theta]$.

For an error $\mathbf{e}$, being small means that $\mathbf{e}$ has low rank weight. For the erasure $\boldsymbol{\varepsilon}$, there are two definitions, which are more involved. In the first part of this Section, we introduce the decoding problem in presence of errors only and show how to solve it by solving the reconstruction problem of θ-polynomials. Then we recall two models of erasures introduced in [Rot91, BM85, LSS14, KK08] adapted to existing applications (line erasures and network coding erasures for instance) and show how the problem of decoding in presence of these errors and erasures can be reduced to the decoding problem of errors in a generalized Gabidulin code derived from the original one.

By choosing a fixed K-basis $\mathcal{B}$ of $K \hookrightarrow L$, we will indifferently consider the vectors of L^n as $m \times n$ matrices over K.

3.1 Decoding errors

We define formally the decoding problem as

Definition 13 (Decoding problem $\mathrm{Dec}(n, k, t, \mathbf{g}, \mathbf{y})$)

- **Input:**
  - $n, k \in \mathbb{N}$, $k \leqslant n$;
  - support $\mathbf{g} = (g_1, \ldots, g_n) \in L^n$, where the g_i's are K-linearly independent;
  - the Gabidulin code $\mathrm{Gab}_{\theta,k}(\mathbf{g})$;
  - $t \in \mathbb{N}$, the rank weight of the error vector (or matrix);
  - $\mathbf{y} = (y_1, \ldots, y_n) \in L^n$.
- **Output:**

- $(\mathbf{c}, \mathbf{e})$ or, equivalently, $(f(X), \mathbf{e})$, such that:
  1. $\mathbf{c} \in \mathrm{Gab}_{\theta,k}(\mathbf{g})$, or, equivalently, $f(X) \in L[X; \theta]$, $\deg f(X) < k$;
  2. $\mathbf{e} = (e_1, \ldots, e_n) \in L^n$ with $w(\mathbf{e}) \leqslant t$;
  3. $\mathbf{y} = \mathbf{c} + \mathbf{e} = f\{\mathbf{g}\} + \mathbf{e}$,
- **or** return **fail** is no such solution exists.

Remark 3 We stated a general definition of the decoding problem. The rank weight t mentioned in the problem is an upper-bound on the rank of the error-vector. We will see in the following that this problem can be solved for $t = \lfloor (d-1)/2 \rfloor$, with at most one solution.

Example 4 In the context of Example 3, suppose we receive

$$
\begin{pmatrix} 1 & -1 & 1 & 3 & -1 & 1 \\ 1 & 0 & -1 & 0 & -1 & -2 \\ 0 & 1 & 0 & -1 & -1 & -1 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & -1 & 3 & 2 & -1 & 0 \\ 0 & 1 & -1 & -1 & -1 & -2 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 & -1 \\ 1 & 0 & 0 & 0 & -1 & -1 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 2 & 0 & -1 & -1 \\ 1 & 0 & 0 & 1 & -1 & -1 \end{pmatrix} + \begin{pmatrix} 1 & -1 & 1 & 2 & 0 & 1 \\ 1 & -1 & -1 & 0 & 0 & -1 \\ -1 & 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & -1 & 1 & 2 & 0 & 1 \\ -1 & 1 & -1 & -2 & 0 & -1 \end{pmatrix}
$$

which can be written in vector form

$$
(y_1, \ldots, y_6) = (c_1, \ldots, c_6) + (\epsilon_1, -\epsilon_1, \epsilon_2, \epsilon_1 + \epsilon_2, 0, \epsilon_2),
$$

where $\epsilon_1 = 1 + \alpha - \alpha^2 + \alpha^4 - \alpha^5$ and $\epsilon_2 = 1 - \alpha + \alpha^3 + \alpha^4 - \alpha^5$. In this case we have an error of rank 2.

Our decoding method is inspired by the so-called Welch-Berlekamp algorithm [BW86, GS92], and we first present an intermediate problem.

Definition 14 (Non Linear Reconstruction problem $\mathrm{NLR}(n, k, t, \mathbf{g}, \mathbf{y})$)

- **Input:**
  - $n, k \in \mathbb{N}$, $k \leqslant n$;
  - $t \in \mathbb{N}$, the rank weight of error ;
  - $\mathbf{g} = (g_1, \ldots, g_n) \in L^n$, where the g_i's are K-linearly independent;
  - $\mathbf{y} = (y_1, \ldots, y_n) \in L^n$.
- **Output:**
  - $f(X), V(X) \in L[X; \theta]$ such that:
    1. $\deg f(X) < k$;
    2. $V(X) \neq 0$ and $\deg V(X) \leqslant t$;
    3. $V\{y_i\} = V \cdot f\{g_i\} = V\{f\{g_i\}\}$, $i = 1, \ldots, n$,
  - **or** return **fail** if no such solution exists.

In the above definition, $V(X)$ plays the role of the locator polynomial of the error: if $\mathbf{y} = \mathbf{g} + \mathbf{e}$, where $w(\mathbf{e}) \leqslant t$, then $V(X)$ vanishes on the K-vector space generated by $\mathbf{e}$.

Proposition 9 *There is a one-to-one correspondence between the solutions of the problem $\mathrm{Dec}(n, k, t, \mathbf{g}, \mathbf{y})$ and those of $\mathrm{NLR}(n, k, t, \mathbf{g}, \mathbf{y})$.*

Proof Let $n, k, t, x\mathbf{g}, \mathbf{y}$ be the parameters of the two statements. If $f(X) \in L[X; \theta]$ and $\mathbf{e} \in L^n$ are solution of $\mathrm{Dec}(n, k, \mathbf{g}, \mathbf{y})$, then $f(X)$ and $V(X) = \mathcal{A}_{\langle e_1, \ldots, e_n \rangle}(X)$ are a solution of $\mathrm{NLR}(n, k, t, \mathbf{g}, \mathbf{y})$. Conversely, if $f(X)$ and $V(X)$ are a solution of $\mathrm{NLR}(n, k, \lfloor \frac{n-k}{2} \rfloor, \mathbf{g}, \mathbf{y})$, then $f(X)$ and $\mathbf{e} = (y_1 - f\{g_1\}, \ldots, y_n - f\{g_n\})$ are a solution of $\mathrm{Dec}(n, k, \mathbf{g}, \mathbf{y})$. $\qquad\qquad\square$

Therefore, to decode errors it is sufficient to solve the Non Linear Reconstruction problem. However since the related equations involve products of unknowns, we would have to solve quadratic systems over L, which is a difficult task. In the case we are interested in errors of weight less than the error-correcting capability, we can "linearize" this problem, i.e. consider terms of degree two as new indeterminates, and the solution to the linearized problem will give the solution to the Non Linear Reconstruction problem.

Definition 15 (Linear Reconstruction problem $\mathrm{LR}(n, k, t, \mathbf{g}, \mathbf{y})$)

- **Input:**
    - $n \in \mathbb{N}$, $k \leqslant n$;
    - $t \in \mathbb{N}$ the rank weight of the error;
    - $\mathbf{g} = (g_1, \ldots, g_n) \in L^n$, where the g_i's are K-linearly independent;
    - $\mathbf{y} = (y_1, \ldots, y_n) \in L^n$;
- **Output:**
    - $N(X), W(X) \in L[X; \theta]$ such that
        1. $W(X) \neq 0$
        2. $\deg W(X) \leqslant t$ and
            - either $\deg N(X) \leqslant k + t - 1$ if $n - k$ is even;
            - or $\deg N(X) \leqslant k + t$ if $n - k$ is odd;
        3. $W\{y_i\} = N\{g_i\}$, $i = 1, \ldots, n$.
    - **or** return **fail** if no such solution exists.

A solution $(V(X), f(X))$ of $\mathrm{NLR}(n, k, t, \mathbf{g}, \mathbf{y})$ gives a solution $(V(X) \cdot f(X), V(X))$ of $\mathrm{LR}(n, k, t, \mathbf{g}, \mathbf{y})$. We have the following converse Proposition:

Theorem 6 *Let* $t \leqslant \lfloor \frac{n-k}{2} \rfloor$. *If there exists a solution* $(V(X), f(X))$ *to the problem* $\mathrm{NLR}(n, k, t, \mathbf{g}, \mathbf{y})$, *then any solution* $(W(X), N(X))$ *of* $\mathrm{LR}(n, k, t, \mathbf{g}, \mathbf{y})$ *satisfies*

$$W(X) \cdot f(X) = N(X).$$

Proof Let $(V(X), f(X))$ be a solution of $\mathrm{NLR}(n, k, t, \mathbf{g}, \mathbf{y})$, with $\deg V(X) \leqslant t$, and $\deg f(X) \leqslant k - 1$ and $V \neq 0$. We set

$$e_i = y_i - f\{g_i\}, \quad i = 1, \ldots, n,$$

and $\mathbf{e} = (e_1, \ldots, e_n)$. For all $i = 1, \ldots, n$, since $V(X)$ is non-zero and $\mathbf{e}$ has rank less than or equal to t:

$$V\{e_i\} = V\{y_i - f\{g_i\}\} = V\{y_i\} - V \cdot f\{g_i\} = 0.$$

Let $(N(X), W(X))$ be a solution of $\mathrm{LR}(n, k, t, \mathbf{g}, \mathbf{y})$, with $W(X) \neq 0$ and $\deg W(X) \leqslant t$. For all $i = 1, \ldots, n$,

$$\begin{aligned} W\{e_i\} &= W\{y_i\} - W\{f\{g_i\}\} \\ &= N\{g_i\} - W\{f\{g_i\}\} \end{aligned}$$

Since the vector $(W \{e_1\}, \ldots W \{e_n\})$ has rank less than or equal to t, there exists $U(X) \neq 0$ with degree less than or equal to t such that, for $i = 1, \ldots, n$,

$$U \{W \{e_i\}\} = U \{N \{g_i\} - W \{f \{g_i\}\}\} = 0.$$

Hence, for $i = 1, \ldots n$,

$$(U(X) \cdot (N(X) - W(X) \cdot f(X))) \{g_i\} = 0.$$

Now since $\deg N(X) \leqslant k + t - 1$ if $n - k$ is even and $\deg N(X) \leqslant k + t$ if $n - k$ is odd, $U(X) \cdot (N(X) - W(X) \cdot f(X))$ is a θ-polynomial of degree less than or equal to $k + 2t - 1$ if $n - k$ is even and $k + 2t$ if $n - k$ is odd. Since $t \leqslant \lfloor \frac{n-k}{2} \rfloor$, in both cases, its degree at most $n - 1$. Moreover it vanishes on n K-linearly independent elements. Therefore

$$U(X) \cdot (N(X) - W(X) \cdot f(X)) = 0$$

Since $L[X; \theta]$ has no zero divisor, we have $N(X) = W(X) \cdot f(X)$.   □

As an immediate consequence, provided $t \leqslant \lfloor (n - k)/2 \rfloor$, any solution $(W(X), N(X))$ of $\mathrm{LR}(n, k, t, \mathbf{g}, \mathbf{y})$ gives the θ-polynomial $f(X)$ solution to $\mathrm{Dec}(n, k, t, \mathbf{g}, \mathbf{y})$, obtained by a simple left Euclidean division $f(X) = W(X) \backslash N(X)$ in the θ-polynomial ring $L[X; \theta]$. This procedure is presented in **Algorithm 1**.

Remark 4 The difference we made between the oddity of $n - k$ is not essential here, but it is convenient to introduce this distinction that will be needed in further algorithms.

Theorem 7 *Let* $\mathbf{g} = (g_1, \ldots, g_n) \in L^n$ *be* K-*linearly independent elements. Let* $\mathbf{y} \in L^n$ *such that* $\mathbf{y} = \mathbf{c}(f) + \mathbf{e}$, *where* $\mathbf{c}(f) \in \mathrm{Gab}_{\theta,k}(\mathbf{g})$ *and such that* $\mathbf{e}$ *has minimum rank weight. If* $w(\mathbf{e}) \leqslant \lfloor \frac{n-k}{2} \rfloor$, *then* **Algorithm 2** *recovers* $(f, \mathbf{e})$ *on inputs* $(n, k, \mathbf{g}, \mathbf{y})$. *If* $w(\mathbf{e}) > \lfloor \frac{n-k}{2} \rfloor$, *then* **Algorithm 2** *returns* **fail**.

Proof Since $w(\mathbf{e}) \leqslant \lfloor \frac{n-k}{2} \rfloor$, then there is a solution to $\mathrm{NLR}(n, k, \lfloor \frac{n-k}{2} \rfloor, \mathbf{g}, \mathbf{y})$, and **Theorem 6** ensures that for $t \leqslant \lfloor \frac{n-k}{2} \rfloor$ any solution $(N(X), W(X))$ to $\mathrm{LR}(n, k, t, \mathbf{g}, \mathbf{y})$ given by **Algorithm 1**, gives a solution to $\mathrm{NLR}(n, k, t, \mathbf{g}, \mathbf{y})$, by a left Euclidean division.

To prove the converse, suppose that **Algorithm 2** returns $(f(X), \mathbf{e})$. This implies that **Algorithm 1** has returned $N(X), W(X)$ with $\deg W(X) \leqslant \lfloor \frac{n-k}{2} \rfloor$; $N(X) = W(X) \cdot f(X)$, where $\deg f(X) \leqslant k$. This last equation gives $W \{f \{g_i\}\} = N \{g_i\} = W \{y_i\}$, *i.e.* $W \{f \{g_i\} - y_i\} = 0$, $1 \leqslant i \leqslant n$, which implies $w(\mathbf{e}) \leqslant \lfloor \frac{n-k}{2} \rfloor$, since $\deg W(X) \leqslant \lfloor \frac{n-k}{2} \rfloor$. Since $\deg f(X) < k$, we have $\mathbf{c} = f \{\mathbf{g}\} \in \mathrm{Gab}_{\theta,k}(\mathbf{g})$. Thus there is a solution to the decoding problem. Therefore, the algorithm returns a failure if and only if the rank of the error vector is larger than $\lfloor \frac{n-k}{2} \rfloor$.   □

Remark 5 Let $t = \lfloor (n - k)/2 \rfloor$ and $s = t - 1$ if $n - k$ is even and $s = t$ otherwise. Let $W(X) = \sum_{i=0}^{t} w_i X^i$, and $N(X) = \sum_{i=0}^{k+s-1} n_i X^i$, then the coefficients w_i's and n_i's of W and N, solution of $\mathrm{LR}(n, k, t, \mathbf{g}, \mathbf{y})$, satisfy

$$\begin{pmatrix} g_1 & \cdots & \theta^{k+s-1}(g_1) & y_1 & \cdots & \theta^t(y_1) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ g_n & \cdots & \theta^{k+s-1}(g_n) & y_n & \cdots & \theta^t(y_n) \end{pmatrix} \cdot \begin{pmatrix} n_0 \\ \vdots \\ n_{k+s-1} \\ -w_0 \\ \vdots \\ -w_t \end{pmatrix} = 0. \tag{8}$$

Algorithm 1 LinearReconstruction (via Gaussian elimination)

Input: $n, k \in \mathbb{N}$, $k \leqslant n$;
$\qquad$ $\mathbf{g} = (g_1, \ldots, g_n) \in L^n$, where the g_i's are K-linearly independent;
$\qquad$ $\mathbf{y} = (y_1, \ldots, y_n) \in L^n$.
Output: $N(X), W(X) \in L[X; \theta]$ solution to $\mathrm{LR}(K, L, n, k, t = \lfloor \frac{n-k}{2} \rfloor, \mathbf{g}, \mathbf{y})$
 1: Let $s = \lfloor (n-k)/2 \rfloor$ if $n - k$ is even and $s = \lfloor (n-k)/2 \rfloor + 1$ otherwise
 2: Compute the kernel V of the linear system (8), by Gaussian elimination
 3: **if** $V \neq \{0\}$ **then**
 4: $\quad$ get a random non zero vector $(n_0, \ldots, n_{k+s-1}, w_0, \ldots, w_t) \in V$
 5: $\quad$ collect the coefficient n_i's in $N(X)$
 6: $\quad$ collect the coefficient w_i's in $W(X)$
 7: $\quad$ return $(N(X), W(X))$
 8: **else**
 9: $\quad$ return **fail**
10: **end if**

It is a system of $k + s + t + 1$ unknowns in n. The right kernel of the system can be found in $O(n^3)$ arithmetic operations in L, by standard Gaussian elimination.

Algorithm 2 Decoding Gabidulin Codes up to half the minimum distance

---

Input: $n, k \in \mathbb{N}$, $k \leqslant n$;
$\qquad$ $\mathbf{g} = (g_1, \ldots, g_n) \in L^n$, where the g_i's are K-linearly independent;
$\qquad$ $\mathbf{y} = (y_1, \ldots, y_n) \in L^n$.
Output: $f(X) \in L[X; \theta]$, $\mathbf{e} \in L^n$
 1: $(N(X), W(X)) \leftarrow \mathrm{LinearReconstruct}(K, L, \theta, n, k, \mathbf{g}, \mathbf{y})$
 2: **if** $\mathrm{LinearReconstruct}(n, k, \mathbf{g}, \mathbf{y})$ returns **fail then**
 3: $\quad$ return **fail**
 4: **else**
 5: $\quad$ $(f(X), R(X)) \longleftarrow \mathrm{LeftEuclideanDivision}(N(X), W(X))$,
 $\qquad$ i.e. $N(X) = W(X)f(X) + R(X)$
 6: $\quad$ **if** $R(X) = 0$ and $\deg f(X) < k$ **then**
 7: $\quad\quad$ return $f(X), \mathbf{y} - f\{\mathbf{g}\}$
 8: $\quad$ **else**
 9: $\quad\quad$ return **fail**
10: $\quad$ **end if**
11: **end if**

3.2 Decoding errors and erasures

The notion of erasures is somewhat difficult to introduce. A first model of "line erasures" [Rot91,BM85] considers the received word as a matrix (all vectors in L^n are expanded into matrices of size $m \times n$ over K by expanding each coefficient over a K-basis $\mathcal{B} = (b_1, \ldots, b_m)$ of L.):

$$\mathbf{Y} = \mathbf{C} + \mathbf{E} + \boldsymbol{\mathcal{E}} \in (K \cup ?)^{m \times n},$$

where $\mathbf{C} \in \mathrm{Gab}_{\theta, k}(\mathbf{g})$, $\mathbf{E}$ is the error-matrix of rank weight $\leqslant t$ and $\boldsymbol{\mathcal{E}}$ is the "erasures" matrix, with "?" being the erasure symbol. Matrices can not be expressed as vectors in this model.

A second model considers network coding erasures [LSS14, KK08]. It is a bit more involved and does not fit with the above short introduction, and shall be introduced later.

3.2.1 Decoding in presence of line erasures

Some coefficients of $\mathbf{Y}$ are erased. We model these erasures by a matrix $\boldsymbol{\mathcal{E}}$ whose coefficients are ? or 0. The ? correspond to erased coefficients, with the convention that for any $x \in L$, $? + x = ?$. Since the rank of a such matrix is not well-defined, its weight is measured by the *term-rank*.

Definition 16 The *term-rank* $w_{tr}(\mathcal{E})$ of a matrix $\mathcal{E}$ is the minimal size of a set S of rows or columns such that any non-zero entry of the matrix $\mathcal{E}$ belong to a row or a column of S. We denote by S_r and S_c the indices of rows and columns of a minimal covering of the erasure matrix, i.e.

$$\boldsymbol{\mathcal{E}}_{i,j} = ? \implies i \in S_r \text{ or } j \in S_c,$$

furthermore, let $s_r = |S_r|$ and $s_c = |S_c|$.

Notice that coverings, even minimal, are not unique. This was originally the metric considered for applications of Gabidulin codes [Rot91].

Example 5 We receive

$$
\begin{pmatrix}
1 & 0 & ? & 2 & 0 & 0 \\
1 & -1 & 0 & 1 & 0 & -1 \\
0 & 1 & ? & -1 & -2 & -1 \\
? & 1 & ? & ? & 0 & -1 \\
1 & -1 & 2 & 1 & 0 & -1 \\
0 & 1 & ? & 0 & -2 & -1
\end{pmatrix}
=
\begin{pmatrix}
0 & 0 & 0 & 1 & -1 & 0 \\
0 & 1 & 0 & 0 & -1 & -1 \\
1 & 0 & 0 & 0 & -1 & -1 \\
0 & 1 & 0 & 0 & 0 & -1 \\
0 & 0 & 2 & 0 & -1 & -1 \\
1 & 0 & 0 & 1 & -1 & -1
\end{pmatrix}
+
\begin{pmatrix}
1 & -1 & 0 & 1 & 1 & 0 \\
1 & -1 & 0 & 1 & 1 & 0 \\
-1 & 1 & 0 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1 & -1 & 0 & 1 & 1 & 0 \\
-1 & 1 & 0 & -1 & -1 & 0
\end{pmatrix}
+
\begin{pmatrix}
0 & 0 & ? & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & ? & 0 & 0 & 0 \\
? & 0 & ? & ? & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & ? & 0 & 0 & 0
\end{pmatrix}
\tag{9}
$$

In this example, the erasure matrix has term-rank 2, since the third column and the fourth row enable to cover all ? symbols.

Definition 17 (Decoding problem with line erasures)

- Input:
  - $\mathrm{Gab}_{\theta,k}(\mathbf{g})$, with parameters $[n, k, d]_r$;
  - $\mathbf{Y} \in \mathcal{M}_{m \times n}(K)$;
  - $\boldsymbol{\mathcal{E}}$ the erasure matrix, whose term-rank is denoted by s.
- Output:
  - $f \in L[X; \theta]$ with $\deg f(X) < k$;
  - $\mathbf{E} \in \mathcal{M}_{m \times n}(K)$ with $w(\mathbf{E}) \leqslant \lfloor \frac{n-k}{2} \rfloor - s$, such that for all i, j, such that $\boldsymbol{\mathcal{E}}_{i,j} \neq ?$,

$$\mathbf{Y}_{i,j} = \mathbf{C}(f)_{i,j} + \mathbf{E}_{i,j}.$$

The procedure for correcting errors and erasures consists of eliminating erasures by reducing the problem of decoding error and erasures in the initial Gabidulin code to decoding errors of a derived Gabidulin code. We show in the following steps how to deal with column erasures first, then with line erasures, and finally decode using the classical model of rank errors.

1. Column erasures: Let $\widetilde{\mathbf{y}} = (y_i)_{i \notin S_c}$, be the received vector punctured on the erased columns. This is a vector of length $n - s_c$ satisfying

$$\widetilde{\mathbf{y}} = \widetilde{\mathbf{c}}(f(X)) + \widetilde{\mathbf{e}} + \widetilde{\boldsymbol{\varepsilon}},$$

where $\tilde{e}$ and $\tilde{\varepsilon}$ are the punctured version of e and ε. Since $\widetilde{\boldsymbol{\varepsilon}}$ has no more column erasures, the problem is thus now reduced to decoding errors and rows erasures in $\mathrm{Gab}_{\theta,k}(\widetilde{\mathbf{g}})$;

2. Row erasures: Let $\mathcal{V}_r(X) = \mathcal{A}_{\langle b_i, i \in S_r \rangle}$ be the θ-polynomial annihilating the elements of $\mathcal{B}$ labeled by S_r. This implies in particular that $\mathcal{V}_r\{\widetilde{\boldsymbol{\varepsilon}}\} = 0$. Therefore, since $\widetilde{\mathbf{c}}(f(X)) = f\{\widetilde{\mathbf{g}}\}$ for some θ-polynomial $f(X)$ of degree less than or equal to $k - 1$, we obtain:

$$\mathcal{V}_r\{\widetilde{\mathbf{y}}\} = \mathcal{V}_r \cdot f\{\widetilde{\mathbf{g}}\} + \mathcal{V}_r\{\widetilde{\mathbf{e}}\}.$$

The problem is now reduced to decoding errors in $\mathrm{Gab}_{\theta, k+s_r}(\widetilde{\mathbf{g}})$.

3. Correcting errors: Since the rank of $\mathcal{V}_r\{\widetilde{\mathbf{e}}\}$ is at most the rank of $\widetilde{\mathbf{e}}$, we can correct the remaining errors by solving $\mathrm{LR}(n - s_c, k + s_r, \lfloor \frac{(n-s_c)-(k+s_r)}{2} \rfloor, \mathbf{g}, \mathbf{y})$ as shown in previous Section. The rank of the new error should by at most the error capability of the new code. After this step, $f(X)$ is recovered by a Euclidean division on the left by $\mathcal{V}_r(X)$ in $L[X; \theta]$.

The procedure is given in **Algorithm 3**.

Theorem 8 *If $2t + s_r + s_c \leqslant n - k$, then the θ-polynomial f can be uniquely recovered by* **Algorithm 3** *on inputs $n, k, S_r, S_c, \widetilde{\mathbf{g}}, \widetilde{\mathbf{y}}, \mathcal{B}$.*

Proof The third step succeeds if the rank weight of the original error is lower or equal to the error correcting capability of the code that we get after steps 1 and 2, which has length $n - s_c$ and dimension $k + s_r$. Thus, decoding succeeds if

$$w(\mathcal{V}_r\{\widetilde{\mathbf{e}}\}) = t' \leqslant \lfloor \frac{(n - s_c) - (k + s_r)}{2} \rfloor$$

i.e. if

$$2t' + s_c + s_r \leqslant n - k.$$

Note that $t' = w(\mathcal{V}_r\{\widetilde{\mathbf{e}}\}) \leqslant w(\mathbf{e}) = t$, thus $2t + s_c + s_r \leqslant n - k$ is sufficient for ensuring successful decoding. In practice, the weight of the error does not decrease when correcting the erasures, hence $t = t'$.

$\square$

3.2.2 Decoding network coding erasures

The erasure model proposed in [LSS14] is related to the problem of correcting errors and erasures in network coding applications of Gabidulin codes. The received vector is seen in matrix form, in the K-basis $\mathcal{B}$:

$$\mathbf{Y} = \mathbf{C}(f) + \mathbf{E} + \widehat{\mathbf{A_r}} \cdot \mathbf{B_r} + \mathbf{A_c} \cdot \widehat{\mathbf{B_c}},$$

where $\widehat{\mathbf{A_r}}$ and $\widehat{\mathbf{B_c}}$ are known to the receiver. Denoting by s_r and s_c the number of row and column erasures and by r the rank of the full error, right hand side matrices have sizes $m \times n$, $m \times n$, $m \times s_r$, $s_r \times n$, $m \times s_c$ and $s_c \times n$.

Algorithm 3 Decoding line erasures

Input: A K-basis $\mathcal{B} = (b_1, \ldots, b_m) \in L^m$ of L
$\qquad \mathbf{g} = (g_1, \ldots, g_n) \in L^n$
$\qquad k$ the dimension of the code
$\qquad \mathbf{y} = (y_1, \ldots, y_n) \in L^n$
$\qquad S_c$ and S_r, sets of distinct labels of size s_c and s_r, such that $s_r + s_c \leqslant n - k$
Output: $f \in L[X; \theta]$
1: $\widetilde{\mathbf{g}} \longleftarrow (g_i, i \notin S_c)$
2: $\widetilde{\mathbf{y}} \longleftarrow (y_i, i \notin S_c)$
3: $\mathcal{V}_r(X) \longleftarrow \mathcal{A}_{\langle b_i, i \in S_r \rangle}$
4: $z_i \longleftarrow \mathcal{V}_r \{y_i\}, \ i \notin S_c$
5: $\widetilde{\mathbf{z}} \longleftarrow (z_i, i \notin S_c)$
6: $F(X) \longleftarrow$ **Algorithm 2**$(n - s_c, k + s_r, \widetilde{\mathbf{g}}, \widetilde{\mathbf{z}})$
7: **return** $f(X) = \mathcal{V}_r(X) \backslash F(X)$

Example 6 Here is an example of a codeword altered by a full error, 1 row erasure and 1 column erasure. The bold-written matrices are known to the receiver.

$$\mathbf{Y} = \begin{pmatrix} 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 & -1 \\ 1 & 0 & 0 & 0 & -1 & -1 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 2 & 0 & -1 & -1 \\ 1 & 0 & 0 & 1 & -1 & -1 \end{pmatrix} + \begin{pmatrix} 1 & -1 & 0 & 1 & 1 & 0 \\ 1 & -1 & 0 & 1 & 1 & 0 \\ -1 & 1 & 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 1 & 1 & 0 \\ -1 & 1 & 0 & -1 & -1 & 0 \end{pmatrix} + \begin{pmatrix} \mathbf{1} \\ \mathbf{-1} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{1} \\ \mathbf{-1} \end{pmatrix} \cdot {}^t\begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ -1 \\ 1 \\ -1 \\ 1 \end{pmatrix} \cdot {}^{\mathbf{t}}\begin{pmatrix} \mathbf{1} \\ \mathbf{0} \\ \mathbf{-1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{pmatrix}.$$

The associated decoding problem is then defined as follows.

Definition 18 (Decoding problem with network coding erasures)

- Input:
  - $\mathrm{Gab}_{\theta,k}(\mathbf{g})$, with parameters $[n, k, d]$;
  - $\mathbf{Y} \in \mathcal{M}_{m \times n}(K)$;
  - $\widehat{\mathbf{A_r}} \in \mathcal{M}_{m \times s_r}(K)$;
  - $\widehat{\mathbf{B_c}} \in \mathcal{M}_{s_c \times n}(K)$.
- Output:
  - $f \in L[X; \theta]$ with $\deg(f) < k$;
  - $\mathbf{E} \in \mathcal{M}_{m \times n}(K)$ with $w(\mathbf{E}) \leqslant \lfloor \frac{n-k}{2} \rfloor - (s_r + s_c)$;
  - $\mathbf{A_c} \in \mathcal{M}_{s_r \times n}(K)$;
  - $\mathbf{B_r} \in \mathcal{M}_{m \times s_c}(K)$;
  such that
  $$\mathbf{Y} = \mathbf{C}(f) + \mathbf{E} + \widehat{\mathbf{A_r}} \cdot \mathbf{B_r} + \mathbf{A_c} \cdot \widehat{\mathbf{B_c}}.$$

The procedure for correcting errors and erasures is the following:

1. Column erasures: let $\mathbf{U}$ be a $n \times n$ matrix of rank n such that $\widehat{\mathbf{B_c}} \cdot \mathbf{U}$ has its $n - s_c$ last columns equal to $\mathbf{0}$. This matrix $\mathbf{U}$ exists and corresponds to the column operations that would be applied to reduce $\widehat{\mathbf{B_c}}$ to a column echelon form by Gaussian elimination. We have
   $$\mathbf{Y} \cdot \mathbf{U} = \mathbf{C} \cdot \mathbf{U} + \mathbf{E} \cdot \mathbf{U} + \widehat{\mathbf{A_r}} \cdot \mathbf{B_r} \cdot \mathbf{U} + \widehat{\mathbf{A_c}} \cdot \mathbf{B_c} \cdot \mathbf{U}.$$

   Let $\widetilde{\mathbf{g}}$ be the $n - s_c$ last positions of vector $\mathbf{g}\mathbf{U}$, $\widetilde{\mathbf{y}}$ be the $n - s_c$ last positions of vector $\mathbf{y}\mathbf{U}$ and $\widetilde{\mathbf{e}}$ the $n - s_c$ last positions of vector $\mathbf{e}\mathbf{U}$. Since puncturing $\widehat{\mathbf{A_c}} \cdot \mathbf{B_c} \cdot \mathbf{U}$ gives 0, the problem is thus now reduced to decoding errors and rows erasures in $\mathrm{Gab}_{\theta,k}(\widetilde{\mathbf{g}})$ with the new "received word" $\widetilde{\mathbf{y}}$.

2. Row erasures: Let $\mathcal{V}_r(X) \in L[X; \theta]$ be the annihilator of the K-vector space generated by the s_r columns of $\widehat{\mathbf{A_r}}$. Then the problem rewritten under vector form becomes

$$\mathcal{V}_r\{\widetilde{\mathbf{y}}\} = \mathcal{V}_r \cdot f\{\widetilde{\mathbf{g}}\} + \mathcal{V}_r\{\widetilde{\mathbf{e}}\}.$$

Since $\mathcal{V}_r(X)$ has degree s_r, the problem is now reduced to decoding errors in the Gabidulin code $\mathrm{Gab}_{\theta, k+s_r}(\widetilde{\mathbf{g}})$;

3. Correcting errors: Since the rank of $\mathcal{V}_r\{\widetilde{\mathbf{e}}\}$ is at most the rank of $\widetilde{\mathbf{e}}$, this can be done by solving $\mathrm{LR}(n - s_c, k + s_r, \lfloor \frac{(n - s_c) - (k + s_r)}{2} \rfloor, \widetilde{\mathbf{g}}, \widetilde{\mathbf{y}})$ as shown in previous Section. The rank of the new error should by at most the error capability of the new code. The output of the error decoding algorithm is actually $\mathcal{V}_r(X) \cdot f(X)$, and $f(X)$ is recovered by a division on the left by $\mathcal{V}_r(X)$.

The procedure is detailed in **Algorithm 4**.

Theorem 9 *If* $2t + s_r + s_c \leqslant n - k$, *then* **Algorithm 4** *uniquely recovers the codeword or the message polynomial, on inputs* $n, k, \mathbf{g}, \mathbf{y}, \widehat{\mathbf{B_c}}, \widehat{\mathbf{A_r}}$, *of size given in the network coding erasures model.*

Proof The proof is the same that in the first erasure model.   □

Algorithm 4 Decoding network coding erasures

Input: $\mathbf{g} = (g_1, \ldots, g_n) \in L^n$
  $\mathbf{y} = (y_1, \ldots, y_n) \in L^n$
  $\widehat{\mathbf{B_c}} \in K^{s_c \times n}$ and $\widehat{\mathbf{A_r}} \in K^{m \times s_r}$,
  k the dimension of the code
  s_r (resp. s_c) the number of row (resp. column) erasures, such that $s_r + s_c \leqslant n - k$
Output: $f(X) \in L[X; \theta]$
 1: Find $\mathbf{U} \neq 0 \in M_n(K)$ such that $\widehat{\mathbf{B_c}} \cdot U$ is zero on the $n - s_c$ last columns
 2: $\widetilde{\mathbf{g}} \longleftarrow (\mathbf{gU})_{[s_c + 1..n]}$
 3: $\widetilde{\mathbf{y}} \longleftarrow (\mathbf{yU})_{[s_c + 1..n]}$
 4: $\mathcal{V}_r(X) \longleftarrow$ annihilator of the column space of $\widehat{\mathbf{A_r}}$
 5: $\widetilde{z}_i \longleftarrow \mathcal{V}_r\{\widetilde{y}_i\}$
 6: $\widetilde{\mathbf{z}} \longleftarrow (z_i)_{[s_c + 1..n]}$
 7: $F(X) \longleftarrow$ **Algorithm 2**$(n - s_c, k + s_r, \widetilde{\mathbf{g}}, \widetilde{\mathbf{z}})$
 8: **return** $f(X) = \mathcal{V}(\mathcal{X})_r \backslash F(X)$

Remark 6 Algorithms dealing with erasures use **Algorithm 2** on codes that are constructed from the original Gabidulin codes. From the original code, the decoder constructs a new shorter with the knowledge of the erasures. Then the errors are decoded in this shorter code. The algorithms have to be adapted to take into account the different erasures. Another option would have been to consider syndrome decoding algorithms working on the same input code as this was done in [LSS14], for instance. For implementation considerations, these algorithms could be more interesting than the ones we propose. We leave these studies for future work.

4 The Welch-Berlekamp like algorithm

The decoding algorithm presented in the previous Section is based upon the resolution of the so-called linear reconstruction problem presented in **Definition 15**. In this Section, we present an algorithm derived from the Welch-Berlekamp one [BW86], in the version formulated by Gemmel and Sudan [GS92], solving this linear reconstruction problem. Originally used for the decoding of Reed-Solomon codes, this algorithm was adapted to Gabidulin codes over finite fields by Loidreau [Loi06]. The version that we present in this Section is a generalization of the latter one. It also works on finite field extensions by replacing the automorphism θ by the Frobenius automorphism. Our version takes into account cases that were not covered by the algorithm in [Loi06].

We first present the algorithm, and secondly, we prove that it indeed solves the linear reconstruction problem. Then we study the complexity of decoding rank errors up to the error-correcting capability, by using this algorithm together with the left Euclidean division of θ-polynomials. We show that it is always quadratic. Finally we present two variants of the decoding and precise their effects on the complexity.

4.1 The Welch-Berlekamp like algorithm

The idea is to compute two pairs (N_0, W_0) and (N_1, W_1) of θ-polynomials which satisfy the interpolation conditions of the problem $\mathrm{LR}(n, k, \lfloor (n-k)/2 \rfloor, \mathbf{g}, \mathbf{y})$ (see **Definition 15**):

$$W\{y_i\} = N\{g_i\}, 1 \leqslant i \leqslant n, \tag{10}$$

and such that at least one of the pairs satisfies the final degree conditions:

$$\deg(N) \leqslant \begin{cases} k + \lfloor \frac{n-k}{2} \rfloor - 1, & \text{if } n - k \text{ even}, \\ k + \lfloor \frac{n-k}{2} \rfloor, & \text{if } n - k \text{ odd}, \end{cases} \tag{11}$$
$$-\infty < \deg(W) \leqslant t.$$

Therefore our aim is to control the growth of the degrees of the respective polynomials, but ensuring that at each round in the loop, the interpolation conditions are satisfied. **Algorithm 5** will be analyzed in the next Section, but we present here the ideas:

1. *Initialization step:* from lines 2 to 7. There are two way of constructing pairs of polynomials of relatively small degree satisfying the interpolation condition at step k.
    - One of the pair is formed with $(N_0 = \mathcal{A}_{<g_1,\ldots,g_k>}, W_0 = 0)$, where $\mathcal{A}_{<g_1,\ldots,g_k>}$ is the annihilator polynomial, see **Definition 6**. In that case we have $\deg(N_0) = k$. Note that this pair does not satisfy the degree conditions since $W_0 = 0$ and $\deg(N_0) = k > k - 1$.
    - The other one is formed with $(N_1 = \mathcal{I}_{[g_1,\ldots,g_k],[y_1,\ldots,y_k]}, W_1 = X)$, where $\mathcal{I}_{[g_1,\ldots,g_k],[y_1,\ldots,y_k]}$, is the interpolating polynomial, see **Definition 7**. In that case $\deg(N_1) = k - 1$ and $\deg(W_1) = 0$, therefore, this pair of polynomials satisfy the degree conditions.
2. *Interpolation step (rounds $k + 1 \leqslant j \leqslant n$):* lines 8 to 45. From two pairs of polynomials satisfying the interpolation condition and such that at least one of them satisfies the degree conditions at round j,

$$W\{y_i\} = N\{g_i\}, 1 \leqslant i \leqslant j,$$

we construct two pairs of polynomials satisfying

$$W\{y_i\} = N\{g_i\}, 1 \leqslant i \leqslant j+1$$

such that at least one of the pairs satisfies the degree conditions at round $j+1$. To ensure this, we evaluate the discrepancy vectors $\mathbf{u} = (u_i)_{i=1}^n$, computing the difference vectors $(N\{g_i\} - W\{y_i\})_{i=1}^n$ at every round of the loop. At round i, it must satisfy

$$\mathbf{u} = (0, \ldots, 0, u_{i+1}, \ldots, u_n)$$

This controls the effectiveness of the interpolation condition. To make sure that at least one of the pairs satisfies the degree conditions we increase the degree of one pair on average by one every two rounds thanks to the updates presented in **Table 1**. In part 4.2 we prove that one of the obtained pairs satisfies the degree requirements (11).

$u_{0,i}$	$u_{1,i}$	A_1'	A_0'	type
$*$	$\neq 0$	$\left(X - \frac{\theta(u_{1,i})}{u_{1,i}}\right) \cdot A_1$	$A_0 - \frac{u_{0,i}}{u_{1,i}} A_1$	1
$= 0$	$= 0$	$X \cdot A_1$	A_0	2
$\neq 0$	$= 0$	no update		3

Table 1 The update formulas depending on the defects. A denotes N or W, which have the same update formula, or $\mathbf{u}_s = (u_{s,i})_i$, for $s = 0, 1$. We denote by A' the update of A, in order to distinguish its value at the beginning and at the end of a round.

4.2 Proof of correctness of the algorithm

First we suppose that we never fall in the case of the *secondary loop* (lines 10 to 21). The easy part is to prove that if the interpolation condition is satisfied at round j, then it is also satisfied at round $j + 1$. As a consequence since the interpolation condition is satisfied by construction at the beginning of *Interpolation step*, it is also satisfied at the end of the algorithm for both pairs of polynomials.

Proposition 10 (Interpolation) *Let $k+1 \leqslant j \leqslant n-1$ such that $W_0\{y_i\} = N_0\{g_i\}$ and $W_1\{y_i\} = N_1\{g_i\}$ for all $1 \leqslant i \leqslant j$. Let (N_0', W_0') and (N_1', W_1') be the polynomials obtained from (N_0, W_0) and (N_1, W_1) from any update described in* **Table 1**. *Then $W_0'\{y_i\} = N_0'\{g_i\}$ and $W_1'\{y_i\} = N_1'\{g_i\}$, for all $1 \leqslant i \leqslant j+1$.*

Proof We only prove for updates of type 1, that is:

1. At round j, $N_1' = \left(X - \frac{\theta(u_{1,j})}{u_{1,j}}\right) \cdot N_1$ and $W_1' = \left(X - \frac{\theta(u_{1,j})}{u_{1,j}}\right) \cdot W_1$:

   For all $1 \leqslant i \leqslant j$, we have $N_1'\{g_i\} = \theta(N_1\{g_i\}) - \frac{\theta(u_{1,j})}{u_{1,j}} N_1\{g_i\}$ and $W_1'\{y_i\} = \theta(W_1\{y_i\}) - \frac{\theta(u_{1,j})}{u_{1,j}} W_1\{y_i\}$. From this we only have to check the equality for $i = j+1$, since for $i \leqslant j$ this comes from the hypotheses of the Theorem. For $i = j+1$ we have $u_{1,j+1} = N_1\{g_{j+1}\} - W_1\{y_{j+1}\}$ and by reordering the terms

   $$N_1'\{g_{j+1}\} = \frac{1}{u_{1,j+1}} \left(N_1\{g_{j+1}\}\theta(W_1\{y_{j+1}\}) - \theta(N_1\{g_{j+1}\})W_1\{y_{j+1}\}\right).$$

   We obtain the same value for $W_1'\{y_{j+1}\}$.

Algorithm 5 Reconstruction algorithm

Input: $k, n \in \mathbb{N}$, $k \leqslant n$
$\quad\quad\quad \mathbf{g} = (g_1, \ldots, g_n) \in L^n$, K-linearly independent elements
$\quad\quad\quad \mathbf{y} = (y_1, \ldots, y_n) \in L^n$
Output: N and W solutions to
 1: # Initialization step
 2: $N_0(X) \longleftarrow \mathcal{A}_{<g_1,\ldots,g_k>}$
 3: $W_0(X) \longleftarrow 0$
 4: $N_1(X) \longleftarrow \mathcal{I}_{[g_1,\ldots,g_k],[y_1,\ldots,y_k]}$
 5: $W_1(X) \longleftarrow 1$
 6: $\mathbf{u}_0 = (u_{0,i})_{i=1}^n \longleftarrow N_0\{\mathbf{g}\} - W_0\{\mathbf{y}\}$
 7: $\mathbf{u}_1 = (u_{1,i})_{i=1}^n \longleftarrow N_1\{\mathbf{g}\} - W_1\{\mathbf{y}\}$
 8:
 9: # Interpolation step
10: **for** i from $k+1$ to n **do**
11: $\quad$ # Secondary loop
12: $\quad j \longleftarrow i$
13: $\quad$ **while** $u_{0,j} \neq 0$ and $u_{1,j} = 0$ and $j \leqslant n$ **do**
14: $\quad\quad j \longleftarrow j+1$
15: $\quad$ **end while**
16: $\quad$ **if** j=n+1 **then**
17: $\quad\quad$ return (N_1, W_1)
18: $\quad$ **else**
19: $\quad\quad$ # Permutation of the components of the positions i and j
20: $\quad\quad i \leftrightarrow j$
21: $\quad$ **end if**
22:
23: $\quad$ # Updates of θ-polynomials, according to discrepancies
24: $\quad$ **if** $u_1 \neq 0$ **then**
25: $\quad\quad N_1' \longleftarrow (X - \frac{\theta(u_{1,i})}{u_{1,i}}) \cdot N_1$
26: $\quad\quad W_1' \longleftarrow (X - \frac{\theta(u_{1,i})}{u_{1,i}}) \cdot W_1$
27: $\quad\quad N_0' \longleftarrow N_0 - \frac{u_{0,i}}{u_{1,i}} N_1$
28: $\quad\quad W_0' \longleftarrow W_0 - \frac{u_{0,i}}{u_{1,i}} W_1$
29: $\quad$ **end if**
30: $\quad$ **if** $u_0 = 0$ and $u_1 = 0$ **then**
31: $\quad\quad N_1' \longleftarrow X \cdot N_1$
32: $\quad\quad W_1' \longleftarrow X \cdot W_1$
33: $\quad\quad N_0' \longleftarrow N_0$
34: $\quad\quad W_0' \longleftarrow W_0$
35: $\quad$ **end if**
36:
37: $\quad N_0 \longleftarrow N_1'$
38: $\quad W_0 \longleftarrow W_1'$
39: $\quad N_1 \longleftarrow N_0'$
40: $\quad W_1 \longleftarrow W_0'$
41:
42: $\quad$ # Discrepancies updates
43: $\quad \mathbf{u}_0 = (u_{0,i})_{i=1}^n \longleftarrow N_0\{\mathbf{g}\} - W_0\{\mathbf{y}\}$
44: $\quad \mathbf{u}_1 = (u_{1,i})_{i=1}^n \longleftarrow N_1\{\mathbf{g}\} - W_1\{\mathbf{y}\}$
45: **end for**
46:
47: return N_1, W_1

---

2. At round j, $N_0' = N_0 - \frac{u_{0,j}}{u_{1,j}} N_1$ and $W_0' = W_0 - \frac{u_{0,j}}{u_{1,j}} W_1$:
   For all $1 \leqslant i \leqslant j$, the equality comes from the hypotheses of the Theorem: $N_0 \{g_i\} = W_0 \{y_i\}$ and $N_1 \{g_i\} = W_1 \{y_i\}$.
   Now since for $i = j + 1$ we have $u_{1,j+1} = N_1 \{g_{j+1}\} - W_1 \{y_{j+1}\}$ and $u_{0,j+1} = N_0 \{g_{j+1}\} - W_0 \{y_{j+1}\}$, we obtain

$$N_0' \{g_{j+1}\} = \frac{1}{u_{1,j+1}} \left(W_0 \{y_{j+1}\} N_1 \{g_{j+1}\} - W_1 \{y_{j+1}\} N_0 \{g_{j+1}\} \right).$$

   The same value is obtained for $W_0' \{y_{j+1}\}$.

For the other updates, the interpolation property is obviously satisfied. □

We have proved that our algorithm correctly interpolated the polynomials at every round in the *Interpolation step*. Now we have to control the degrees to check that at least one of the pairs of polynomials satisfies the degree conditions (11).

First we give an upper bound on the degrees of the polynomials in the algorithm.

Proposition 11 (Degree Control) *At the end of round j, $k + 1 \leqslant j \leqslant n$ of the Interpolation step in* **Algorithm 5**, *the degrees of the polynomials satisfy:*

$$\begin{aligned}
\deg(N_0) &\leqslant k + \lfloor \tfrac{j-k}{2} \rfloor, & \deg(W_0) &\leqslant \lfloor \tfrac{j-k+1}{2} \rfloor, \\
\deg(N_1) &\leqslant k - 1 + \lfloor \tfrac{j-k+1}{2} \rfloor, & \deg(W_1) &\leqslant \lfloor \tfrac{j-k}{2} \rfloor.
\end{aligned}$$

Proof The proof is made by induction. At the beginning of round $j = k + 1$, N_0 and N_1 are respectively the annihilator of degree k and the interpolating polynomial of degree $k - 1$. Moreover $W_0 = 0$ and $W_1 = X$. Therefore, by considering the updates in **Table 1** and the fact that the polynomials are swapped at the end of round j (lines 37 to 40), at the end of the round 1, we have:

- $\deg(N_0) = k$, $\deg(W_0) = 1$, since polynomials are multiplied by an affine polynomial, increasing thus their degrees exactly by one.
- $\deg(N_1) = k$, since it is the sum of a polynomial of degree exactly k and a polynomial of degree strictly less than k.
- $\deg(W_1) \leqslant 1$. There is no certainty on the exact degree of W_1 since it is the sum of 0 and a constant, possibly 0.

Suppose that the property is true for some $k + 1 \leqslant j \leqslant n - 1$. There are two cases:

1. $j - k = 2u$ is even.
   By hypothesis, at the beginning of round $j + 1$ (corresponds to the end of round j), we have $\deg(N_0) \leqslant k + u$, $\deg(W_0) \leqslant u$, $\deg(N_1) \leqslant k - 1 + u$, $\deg(W_1) \leqslant u$. At the end of round $j + 1$ we have therefore,

$$\begin{aligned}
\deg(N_0) &\leqslant k + u, & \deg(W_0) &\leqslant u + 1, \\
\deg(N_1) &\leqslant k - 1 + u + 1, & \deg(W_1) &\leqslant u.
\end{aligned}$$

   Since $j - k = 2u$ is even, $\lfloor \frac{j+1-k}{2} \rfloor = u$, and $\lfloor \frac{j+1-k+1}{2} \rfloor = u + 1$.

2. $j - k = 2u + 1$ is odd.
   By hypothesis, at the beginning of round $j + 1$ $\deg(N_0) \leqslant k + u$, $\deg(W_0) \leqslant u + 1$, $\deg(N_1) \leqslant k + u$, $\deg(W_1) \leqslant u$. At the end of round $j + 1$,

$$\begin{aligned}
\deg(N_0) &\leqslant k + u + 1, & \deg(W_0) &\leqslant u + 1, \\
\deg(N_1) &\leqslant k - 1 + u + 1, & \deg(W_0) &\leqslant u + 1.
\end{aligned}$$

   Since $j - k = 2u + 1$, we have: $\lfloor \frac{j+1-k+1}{2} \rfloor = \lfloor \frac{j+1-k}{2} \rfloor = u + 1$.

Now suppose that the upper bound on the degrees is true for some $k+1 \leqslant j \leqslant n-1$. Then the chosen updates show that it is still true for $j + 1$. $\qquad\square$

So far this Proposition gives upper bounds, but does not ascertain that at the end of the algorithm we will not fall into a degenerated case ($W_1 = 0$). To this end we will use the following Proposition which shows that at every round at least one polynomial of every pair reaches the degree upper bound of the previous Proposition.

Proposition 12 *At the end of loop* $k + 1 \leqslant j \leqslant n$ *in* **Algorithm 5***:*

- *If* $j - k = 2u + 1$, *then* $\deg(N_1) = k + u$ *and* $\deg(W_0) = u + 1$;
- *If* $j - k = 2u$, *then* $\deg(N_0) = k + u$ *and* $\deg(W_1) = u$.

Proof Let $\mathcal{P}_j$ for all $k \leqslant j \leqslant n - 1$ the property:

- $\deg(N_1) = k + u$, $\deg(W_0) = u + 1$, if $j = k + 2u + 1$.
- $\deg(N_0) = k + u$, $\deg(W_1) = u$, if $j = k + 2u$.

From the initialization round in the proof of **Proposition 11**, $\mathcal{P}_{k+1}$ is satisfied. Suppose now that $\mathcal{P}_{k+2u+1}$ is satisfied, we show that $\mathcal{P}_{k+2u+2}$ is satisfied.

Combining the induction property and the upper bounds of **Proposition 11**, at the end of round $j = k + 2u + 1$, we have:

$$\deg(N_0) \leqslant k + u, \quad \deg(W_0) = u + 1,$$
$$\deg(N_1) = k + u, \quad \deg(W_1) \leqslant u.$$

Hence at the beginning of round $j + 1 = k + 2u + 2$, the same bounds and equalities hold.

Since $\deg(W_0) > \deg(W_1)$ the updates in the loop show that at the end of the round $k + 2u + 2$, $\deg(W_1) = u + 1$. The polynomial N_0 is obtained from N_1 by a left multiplication by an affine monic polynomial. Hence $\deg(N_0) = k + u + 1$ at the end of round $k + 2u + 2$. The upper bounds come from **Proposition 11**. Hence $\mathcal{P}_{k+2u+2}$ is satisfied.

By using the same arguments we show that if $\mathcal{P}_{k+2u}$ is satisfied then $\mathcal{P}_{k+2u+1}$ is also satisfied. Since we proved that $\mathcal{P}_{k+1}$ was satisfied, by induction we proved that $\mathcal{P}_j$ is satisfied for all $k + 1 \leqslant j \leqslant n$. $\qquad\square$

A direct consequence of this Proposition is that $W = 0$ can never occur for any pair of polynomials in the *Interpolation step*. Namely, this would imply that the corresponding polynomial N satisfies $N\{g_i\} = 0$, for all $1 \leqslant i \leqslant j$. However, $\deg(N) \leqslant k + \lfloor \frac{j-k}{2} \rfloor < j$, for $j \geqslant k + 1$. Since the g_i's are linearly independent by the hypotheses on the input of the algorithm, this implies that $N = 0$. But from **Proposition 12**, it is not possible to have the two polynomials of the pair which do not reach the upper bound on the degree.

This is the reason why the algorithm returns the pair of polynomials of smallest degree. Now by combining all the previous results, we obtain:

Theorem 10 (Proof of correctness of the algorithm) *Let*

- $K \hookrightarrow L$ *be a cyclic field extension;*
- θ *be a generator of its automorphism group* $\mathrm{Aut}_K(L)$;
- $\mathbf{g} = (g_1, \ldots, g_n) \in L^n$ *be* K*-linearly independent elements;*
- $\mathbf{y} = (y_1, \ldots, y_n) \in L^n$;

- *an integer $k \leqslant n \in \mathbb{N}$*

Then the pair (N_1, W_1) returned by **Algorithm 5** *is a solution of* $\mathrm{LR}(n, k, t = \lfloor (n - k)/2 \rfloor, \mathbf{g}, \mathbf{y})$.

Proof **Proposition 10** shows that the returned pair satisfies the interpolation step for all $j = 1, \ldots, n$. Let $t = \lfloor (n - k)/2 \rfloor$. From **Proposition 11** the degrees of the returned pair (N_1, W_1) satisfy

$$\deg(N_1) \leqslant \begin{cases} k - 1 + t, & \text{if } n - k = 2t \\ k + t, & \text{if } n - k = 2t + 1 \end{cases}$$
$$\deg(W_1) \leqslant t$$

and **Proposition 12** ensures that $W_1 \neq 0$, therefore from (11) the degree conditions are satisfied. □

Now we deal with the case of what occurs in the case where the *secondary loop* (lines 10 to 21) is activated. In the case where at round i, $u_{1,i} = 0$ and $u_{0,i} \neq 0$, we search for the first position $i < s \leqslant n$ such that either $u_{1,s} \neq 0$ or $u_{1,s} = 0 = u_{0,s}$. There are two cases :

- Either there exists some $i < s \leqslant n$ satisfying either $u_{1,s} \neq 0$ or $u_{1,s} = 0 = u_{0,s}$. In that case the positions i and s are exchanged. This corresponds to a permutation of one position along all the input vectors and has no impact on the interpolation and degree conditions.
- Or such an s does not exist and this means that the pair of polynomials (N_1, W_1) at the beginning of round i satisfies the interpolation conditions (the discrepancy vector $\mathbf{u}_1$ is equal to $\mathbf{0}$), and from **Proposition 11** and **Proposition 12** the degree conditions are satisfied.

4.3 Complexity of the decoding

The complexity of the decoding procedure consists of adding

- The complexity of **Algorithm 5** returning the θ-polynomials (N_1, W_1);
- The complexity of the left Euclidean division of N_1 by W_1.

4.3.1 Prerequisite

Before studying the complexity of the **Algorithm 5**, we give the complexity of elementary functions used. We count the number of additions in L, multiplications in L, uses of θ and divisions in L. For certain field extensions $K \hookrightarrow L$, an element of L is represented by its coefficients in a K-basis of L, and the automorphism consists in permuting these coefficients. So uses of θ do not need computations in these cases, for example in Kummer or cyclotomic extensions.

Let $A, B \in L[X; \theta]$ be θ-polynomials of degrees a and b, and let $x \in L$. The complexity of arithmetic operations is given in the following table.

operation	additions	multiplications	uses of θ	divisions
$A + B$	$1 + \min(a,b)$	0	0	0
$A \cdot B$	ab	$(1+a)(1+b)$	$a(1+b)$	0
$A\{x\}$	a	$a+1$	a	0
Euclidean division $A = B \cdot Q + R$	$(a-b)b$	$(a-b)(b+1)$	$(a-b)(2b)$	$(a-b)$

The multiplication algorithm is the naive one, but in our case it is optimal since we only multiply by polynomials of degree 1.

4.3.2 Error correction

At the *Initialization step*, we have to compute the annihilator and the interpolating polynomials. They can be simultaneously computed with **Algorithm 6**.

Algorithm 6 Annihilator and Interpolation polynomials

Input: $g_1, \ldots, g_k \in L$ K-linearly independent
$\qquad$ $y_1, \ldots, y_k \in L$
Output: $\mathcal{A}(X)$, $\mathcal{I}(X)$
1: $\mathcal{A} := 1$
2: $\mathcal{I} := 0$
3: **for** $1 \leqslant i \leqslant k$ **do**
4: $\quad \mathcal{I} := \mathcal{I} + \frac{y_i - \mathcal{I}\{g_i\}}{\mathcal{A}\{g_i\}} \cdot \mathcal{A}$
5: $\quad \mathcal{A} := (X - \frac{\theta(\mathcal{A}\{g_i\})}{\mathcal{A}\{g_i\}}) \cdot \mathcal{A}$
6: **end for**
7: **return** $\mathcal{A}, \mathcal{I}$

Additionally we have to compute the discrepancy vectors:

$$\mathbf{u}_0 = (0, \ldots, 0, \mathcal{A}\{g_{k+1}\}, \ldots, \mathcal{A}\{g_n\})$$
$$\mathbf{u}_1 = (0, \ldots, 0, \mathcal{I}\{g_{k+1}\} - y_{k+1}, \ldots, \mathcal{I}\{g_n\} - y_n)$$

Therefore the complexity of the *Initialization step* is:

	additions	multiplications	uses of θ	divisions
Algorithm 6	$2k^2 - 2k$	$2k^2 - k$	$1.5k^2 - 0.5k$	$2k$
$\mathbf{u}_0$ and $\mathbf{u}_1$	$(2k-1)(n-k)$	$(2k+1)(n-k)$	$(k-1)(n-k)$	0

In the *Interpolation step* the number of arithmetic operations can be upper bounded by considering only updates of type 1. The involved operations consist of:

1. updates of (N_0, W_0) and (N_1, W_1) at round $j = k + i$:
   From **Proposition 11** the degrees of the θ-polynomials N_0, N_1, W_0 and W_1 are respectively less or equal to $k + \lfloor \frac{i-1}{2} \rfloor$, $k - 1 + \lfloor \frac{i}{2} \rfloor$, $\lfloor \frac{i}{2} \rfloor$ and $\lfloor \frac{i-1}{2} \rfloor$.
2. updates of the discrepancy vectors $\mathbf{u}_0$ and $\mathbf{u}_1$. The discrepancy vectors $\mathbf{u}_0$, $\mathbf{u}_1$ at round $j + 1$ in the loop can be obtained from $\mathbf{u}_0$, $\mathbf{u}_1$ at round j by performing the

same updates on the vectors than for the corresponding pairs of polynomials. For instance if we consider updates of type 1:

$$\mathbf{u'}_1 \longleftarrow \theta(\mathbf{u}_1) - \frac{\theta(u_{1,j})}{u_{1,j}}\mathbf{u}_1,$$
$$\mathbf{u'}_0 \longleftarrow \mathbf{u}_0 - \frac{u_{0,j}}{u_{1,j}}\mathbf{u}_1,$$

where θ acts on $\mathbf{u}_1$ component by component.

Therefore, at round j, an upper bound on the complexities is:

	additions	multiplications	uses of θ	divisions
Up. N_0, N_1, W_0 and W_1	$2j-1$	$2j-1$	$j+1$	2
Up. $\mathbf{u}_0$ and $\mathbf{u}_1$	$2(n-j)$	$2(n-j)$	$n-j$	0
Total	$2n-1$	$2n-1$	$n+1$	2

To obtain the full cost one has to sum the complexities for $j = k+1, \dots, n$. Finally to complete the decoding complexity analysis it remains to evaluate the complexity of the final left Euclidean division of N_1 by W_1 which is:

	additions	multiplications	uses of θ	divisions
Left division	$(k-1)\frac{n-k}{2}$	$(k-1)\frac{n-k}{2}$	$(n-k)(k-1)$	0

Since the *secondary loop* consists only on tests and permutations, it has no effect on the complexity. All these evaluation lead to the following Theorem:

Theorem 11 (Decoding complexity) *The complexity of* **Algorithm 5** *for solving the problem* $\mathrm{Dec}(K, L, \theta, n, k, \lfloor \frac{n-k}{2} \rfloor, \mathbf{g}, \mathbf{y})$ *is* $O(n^2)$ *operation in L. More precisely the number of different field operations is upper-bounded by:*

- $2n^2 - 2n + (k-1)\frac{n-k}{2}$ *additions in L,*
- $2n^2 - k + (k-1)(\frac{n-k}{2})$ *multiplications in L,*
- $n^2 + 0.5k^2 - 2n + 1.5k^2 + (n-k)(k-1)$ *uses of θ,*
- $2n$ *divisions in L.*

Remark 7 Since $k(n-k) \leqslant \frac{n^2}{4}$, the total number of multiplication is upper-bounded by $2.125n^2$.

4.3.3 Complexity of the errors and erasures correction

Correcting errors and erasures corresponds to remove the erasures and then correcting the residual rank errors. As seen at Section 3.2, this corresponds to:

- Removing the column erasures by either directly puncturing the columns or computing some Gaussian elimination and then puncturing;
- Removing row erasures by computing $\mathcal{V}_r$ and evaluate it on $(n - s_c)$ elements of L;
- Decoding errors in a generalized Gabidulin code of length $n - s_c$ and dimension $k + s_r$;
- Divide the obtained solutions by $\mathcal{V}_r$.

  Overall a direct corollary of the previous Theorem is

Corollary 2 *Given a generalized Gabidulin code of parameters $[n, k, d]$ and a received word $\mathbf{y}$ with s_r row erasures and s_c column erasures and the maximal number of full-errors, the complexity of recovering the information polynomial f is:*

- $O(n s_c m)$ *operations in K;*
- $O(n^2)$ *multiplications in L.*

4.4 Some improvements

In this Section we present two ways to improve the complexity of **Algorithm 5**. First, we design a variant without divisions in L. This can be of interest since although in finite fields, the complexity of multiplication and division is roughly identical, this is not necessarily the case in other fields.

Second, we use the property that the polynomials N_0 and N_1 lie by construction in the left module generated by the annihilator and the interpolating polynomials computed at the *Initialization step*. This enables to reduce significantly the cost of the updates by updating polynomials of smaller degrees.

4.4.1 A division-free variant

As we will see in a next Section by computing in integer rings of rational fields, it can be very interesting to process the algorithm without making divisions, so that when inputs are integer values, there is no fraction along the computation. This can be accomplished modifying **Algorithm 5** to avoid divisions (see Table 4.4.1).

- Consider the polynomials $\mathcal{A}$, $\mathcal{I}$ returned by **Algorithm 7** on the first k positions of the input vectors of **Algorithm 5**. We have
  - $\mathcal{A} = \mu \, \mathcal{A}_{<g_1,\ldots,g_k>}.$
  - $\mathcal{I} = \lambda \, \mathcal{I}_{[g_1,\ldots,g_k],[y_1,\ldots,y_k]}.$
  Both polynomials are computed without divisions. It is not difficult to see that $\lambda \neq 0$ and $\mu \neq 0$. Therefore, by replacing lines 4 and 5 of **Algorithm 5** by $N_1(X) \longleftarrow \mathcal{I}$ and $W_1(X) \longleftarrow \lambda$, we obtain polynomials still satisfying the interpolation conditions and of the same degree.
- Concerning the interpolation step of **Algorithm 5**, we modify the updates so that there is no more divisions by u_1:

$u_{0,i}$	$u_{1,i}$	A_1'	A_0'	type
$*$	$\neq 0$	$(u_{1,i}X - \theta(u_{1,i})) \cdot A_1$	$u_{1,i}A_0 - u_{0,i}A_1$	1
$= 0$	$= 0$	$X \cdot A_1$	A_0	2
$\neq 0$	$= 0$	no update		3

Table 2 Division-free updates

Since W_1 is not monic anymore, this requires more multiplications. Concerning the Euclidean division $W_1 \backslash N_1$, since W_1 is not monic, this amounts to $k-1$ additional divisions, but these are exact divisions.

Hence this division-free variant of **Algorithm 5** requires roughly 1.5 times more multiplications in L.

4.4.2 Polynomials with lower degree

In the algorithm N_0 and N_1 are updated using additions and left-multiplications. Therefore they lie in the left-module generated by $\mathcal{A}$ and $\mathcal{I}$. This implies that at every

---

Algorithm 7 Annihilator and Interpolation polynomials (Division-free variant)

Input: $g_1, \ldots, g_k \in L$ K-linearly independent
$\qquad$ $y_1, \ldots, y_k \in L$
Output: $\mathcal{A}(X)$ such that $\mathcal{A}\{x_i\} = 0$
$\qquad$ $\mathcal{I}(X)$ and λ such that $\mathcal{I}\{x_i\} = \lambda y_i$
$\quad$ $\mathcal{A} \longleftarrow 1$
$\quad$ $\mathcal{I} \longleftarrow 0$
$\quad$ $\lambda \longleftarrow 1$
$\quad$ **for** $1 \leqslant i \leqslant k$ **do**
$\qquad$ $\mathcal{I} \longleftarrow \mathcal{A}\{x_i\} \cdot \mathcal{I} + (\lambda y_i - \mathcal{I}\{x_i\}) \cdot \mathcal{A}$
$\qquad$ $\lambda \longleftarrow \mathcal{A}\{x_i\} \cdot \lambda$
$\qquad$ $\mathcal{A} \longleftarrow (\mathcal{A}\{x_i\} X - \theta(\mathcal{A}\{x_i\})) \cdot \mathcal{A}$
$\quad$ **end for**
$\quad$ **return** $\mathcal{A}, \mathcal{I}, \lambda$

---

round $k + 1 \leqslant j \leqslant n$ they can be expressed under the form

$$N_i = P_i \cdot \mathcal{A} + Q_i \cdot \mathcal{I}, \ i = 0, 1,$$

for some polynomials P_i and Q_i, which are updated similarly to the corresponding polynomial N_i. Moreover, the polynomials Q_i are initialized by $Q_1 = W_1 = 1$ and $Q_0 = W_0 = 0$. Since the polynomials have the same initialization and the same update, they are equal, that is at every round we have:

$$N_i = P_i \cdot \mathcal{A} + W_i \cdot \mathcal{I}, \ i = 0, 1.$$

If we replace the lines 25 and 27 in the algorithm with

$$P_1' \longleftarrow \left(X - \frac{\theta(u_{1,i})}{u_{1,i}}\right) \cdot P_1,$$
$$P_0' \longleftarrow P_0 - \frac{u_{0,i}}{u_{1,i}} P_1,$$

and lines 31 and 33 accordingly, we now update W_i and P_i, such that $\deg P_i = \deg(N_i) - k$. The discrepancy vectors are updated as before. Hence the number of operations at round j is now upper bounded by

	additions	multiplications	uses of θ	divisions
$\mathbf{u}_0$ and $\mathbf{u}_1$	$2(n - j)$	$2(n - j)$	$n - j + 1$	0
Up. (P_j and W_j)	$2j - k - 1$	$2j - k - 1$	$j - k + 1$	2

The gain in arithmetic operations for the interpolation step is $k(n - k)$ The final division is modified and becomes:

$$W_1 \backslash (P_1 \mathcal{A}) + \mathcal{I}.$$

Since the polynomials in the division have the same degree as in the basic algorithm, the complexity is the same. We need $k + 1$ additional operations to modify the final division. However this effect is largely compensated by the fact that the complexity of computing the P_i's is much smaller than the complexity of computing the N_i's.

Theorem 12 (Complexity improvement) *If one consider using the polynomials P_i rather than N_i in* **Algorithm 5**, *the number of arithmetic operations over the field L is reduced by $k(n - k)$.*

5 The case of number fields

When the code alphabet is an infinite field, like a number field, the proposed decoding algorithm has the disadvantage that the bit-size of intermediate coefficients grows a lot, in such a way that the decoding algorithm is not practical at all. In computer algebra, a standard way of circumventing this problem is to perform computations modulo a large enough prime. In the Section, we discuss this technique in the number field case, and relate a generalized Gabidulin code built with integral elements to its reduction modulo a prime, which turns out to be a classical Gabidulin code over a finite field.

Assuming that the receiver knows an *a priori* bound on the size of the message, or on the size of the error, it can apply this technique with a large enough prime, to get the exact result over the number field, only by doing computations modulo the chosen prime.

5.1 Basic algebraic number theory

In this Section, we recall some definitions and properties about number fields, their integer rings, ideals and ramification. We refer the reader to [Coh93, §4.8.1 and §4.8.2].

Let $\mathbb{Q} \hookrightarrow F$ be a number field of degree $[F : \mathbb{Q}] = m$, and denote by $\mathcal{O}_F$ its *integer ring*. The integer rings of number fields are *Dedekind rings*: a non zero ideal is prime if and only if it is maximal. A prime ideal of $\mathcal{O}_F$ restricted to $\mathbb{Z}$ is a prime ideal $p\mathbb{Z}$ of $\mathbb{Z}$. Conversely, let $p\mathbb{Z}$ be a prime ideal of $\mathbb{Z}$. The ideal $\mathfrak{p}$ generated by $p\mathbb{Z}$ in F is generally not a prime ideal, and we have the following decomposition into prime ideals:

$$\mathfrak{p} = \prod_{i=1}^{q} \mathfrak{p}_i^{e_i} \tag{12}$$

where the $\mathfrak{p}_i$'s are the prime ideals of $\mathcal{O}_F$ whose restriction to $\mathbb{Z}$ is $p\mathbb{Z}$. The exponent e_i is called the *ramification index*. The *residue field* $\mathcal{O}_F/\mathfrak{p}_i$ is an extension of $\mathbb{Z}/p\mathbb{Z}$, its extension degree f_i is called the *residual degree*. We say that $\mathfrak{p}_i$ is *above* $\mathfrak{p}$, and conversely that $\mathfrak{p}$ is *below* $\mathfrak{p}_i$,

We have the relation $\sum_{i=1}^{q} f_i e_i = m$ and when $\mathbb{Q} \hookrightarrow F$ is a Galois extension, we have that all the indices e_i are equal to the same e, and all residual degrees f_i are equal to the same f, with $qfe = m$.

We say that p is *ramified* if there is some i such that $e_i > 1$. It happens only for a finite number of prime numbers p. The number p is *inert* if it is unramified and $q = 1$ (thus $f = m$). Conversely, p *splits totally* in $\mathbb{Z}$ if it is unramified and $q = m$ (thus $e_i = f_i = 1$).

Assume that $F = \mathbb{Q}[z]$ where z is an algebraic integer. Let $T(X) \in \mathbb{Q}[X]$ be the minimal polynomial of z. Then for any prime which doesn't divide the index $[\mathcal{O}_F : \mathbb{Z}[z]]$, we can obtain the prime decomposition of $\mathfrak{p}$ from the decomposition of $T(X)$ modulo p. In particular, the prime p is inert if $T(X)$ is irreducible modulo p [Coh93, Thm 4.8.13].

Example 7 In the ring $\mathbb{Z}[i]$,

- 2 is ramified. Indeed, $2 = (1 + i)(1 - i)$ but $i + 1$ and $i - 1 = i(i + 1)$ span the same ideal.
- 3 is inert, since it is a prime number in $\mathbb{Z}[i]$. Thus, $\mathbb{Z}[i]/(3) \simeq \mathbb{F}_9$.

- 5 splits totally. Indeed, $5 = (2+i)(2-i)$ and these factors span two distinct ideals. Thus, $\mathbb{Z}[i]/(5) \simeq \mathbb{F}_5 \times \mathbb{F}_5$.

Actually, to build relevant examples of generalized Gabidulin codes, we shall need a more elaborate situation, with a base field being an extension of $\mathbb{Q}$. Let $\mathbb{Q} \hookrightarrow K \hookrightarrow L$ be an extension of number fields, $p \in \mathbb{N}$ a prime number, $\mathfrak{p}$ a prime ideal of $\mathcal{O}_K$ above p, let $\mathcal{O}_L$ be the ring of the integers of L, and $\mathfrak{P}$ a prime ideal of $\mathcal{O}_L$ above $\mathfrak{p}$, i.e. $\mathfrak{p} \in \mathfrak{P}$. Then $\mathcal{O}_L/\mathfrak{P}$ is finite degree extension of $\mathcal{O}_K/\mathfrak{p}$. We can describe the Galois group of $\mathcal{O}_K/\mathfrak{p} \hookrightarrow \mathcal{O}_L/\mathfrak{P}$ from the Galois group $\mathrm{Aut}_K(L)$. Recall that the *decomposition group* of $\mathfrak{P}$ is the following subgroup of the Galois group:

$$D_{\mathfrak{P}} = \{\theta \in \mathrm{Aut}_K(L) : \theta(\mathfrak{P}) = \mathfrak{P}\}$$

whose cardinal is ef [Coh93]. Consider the map

$$\psi_{\mathfrak{P}} : D_{\mathfrak{P}} \to \mathrm{Aut}_{\mathcal{O}_K/\mathfrak{p}}(\mathcal{O}_L/\mathfrak{P}) \tag{13}$$
$$\theta \mapsto \overline{\theta}$$

where $\overline{\theta}$ is as follows:

$$\overline{\theta} : \mathcal{O}_L/\mathfrak{P} \to \mathcal{O}_L/\mathfrak{P} \tag{14}$$
$$x + \mathfrak{P} \mapsto \theta(x) + \mathfrak{P}$$

Then $\psi_{\mathfrak{P}}$ is a morphism of groups with kernel

$$I_{\mathfrak{P}} = \left\{\theta \in D_{\mathfrak{P}} : \forall x \in \mathcal{O}_L, \theta(x) - x \in \mathfrak{P}\right\}$$

which is the *inertia group* of $\mathfrak{P}$.

Proposition 13 *[Sam71, §6.2] With the previous notation,*

1. *$\mathcal{O}_L/\mathfrak{P}$ is a Galois extension of $\mathcal{O}_K/\mathfrak{p}$, of degree f,*
2. *$\psi_{\mathfrak{P}}$ is a surjective morphism from $D_{\mathfrak{P}}$ to the Galois group $\mathrm{Aut}_{\mathcal{O}_K/\mathfrak{p}}(\mathcal{O}_L/\mathfrak{P})$,*
3. *$I_{\mathfrak{P}}$ has cardinal e.*

5.2 Integral codes

We now can give some definitions about codes with integer coefficients. Let $\mathbb{Q} \hookrightarrow K \hookrightarrow L$ be a number field and $\mathcal{O}_L$ its integer ring (also called maximal order). We also consider an integral basis $\mathcal{B} = (b_1, \ldots, b_\ell)$ of $\mathcal{O}_L$ (see [Coh93, §4.4]). An integral basis is a basis of the $\mathbb{Z}$-module $\mathcal{O}_L$, and any element $x \in \mathcal{O}_L$ can be uniquely decomposed as

$$x = \sum_{i=1}^{\ell} x_i b_i, x_i \in \mathbb{Z}.$$

For any element $x \in L$, there exists $\lambda \in \mathbb{Z}$ such that $\lambda x \in \mathcal{O}_L$. Let $\mathcal{C} \subset L^n$ be a code with generating matrix $G \in \mathcal{M}_{k,n}(L)$. Then $\mathcal{C}$ admits a generating matrix $G' \in \mathcal{M}_{k,n}(\mathcal{O}_L)$, with $\lambda \in \mathbb{Z}$ such that $\lambda G = G'$. For Gabidulin codes, we will furthermore consider a support $\mathbf{g}$ of integral elements.

Proposition 1 *Let $\mathbb{Q} \hookrightarrow K \hookrightarrow L$ be an extension of number fields, and $\mathcal{O}_K$, $\mathcal{O}_L$ the corresponding integers rings. Let $\mathbf{g} \in L^n$ be a vector of K-linearly independent elements of L. Let $\theta \in \mathrm{Aut}_K(L)$, $0 \leqslant k \leqslant n$, and $\mathcal{C}$ be the code $\mathcal{C} = \mathrm{Gab}_{\theta,k}(\mathbf{g})$. Then there exists a support $\mathbf{g}' \in \mathcal{O}_L^n$ such that $C = \mathrm{Gab}_{\theta,k}(\mathbf{g}')$.*

Proof Let $\lambda \in \mathbb{Z}$ such that $\lambda g_i \in \mathcal{O}_L$, $i = 1, \ldots, n$. Such a λ exists since we have a finite number of g_i's. Then $\mathbf{g}' = \lambda \mathbf{g} \in \mathcal{O}_L^n$ is an integral support which defines the same code.   □

Definition 19 Let $\mathbb{Q} \hookrightarrow K \hookrightarrow L$ be a number field and $\mathcal{O}_L$ its integer ring (also called maximal order). Let $\mathcal{C}$ be a code in L^n. Its associated *integral code* is $\mathcal{O}_{\mathcal{C}} = \mathcal{C} \cap \mathcal{O}_L^n$.

Definition 20 Let $Q \hookrightarrow K \hookrightarrow L$ be an extension of number fields, and let C be Gabidulin code with an integral support $\mathbf{g} = (g_1, \ldots, g_n) \in \mathcal{O}_L^n$, its *restricted code* is

$$\mathcal{G} = \{(f\{g_1\}, \ldots, f\{g_n\}) \ : \ f \in \mathcal{O}_L[X; \theta], \deg(f) < k\}.$$

Remark 8 The restricted code is included in the integral code of the Gabidulin code. Nevertheless, the inclusion is strict. Consider for example a code whose support is $(2, 2\alpha, 2\alpha^2, \ldots)$, and the information word $f = \frac{1}{2}X^0$. The corresponding codeword is $(1, \alpha, \alpha^2, \ldots)$, which belongs to the integral code but not to the restricted code.

5.3 The intermediate growth of coefficients

Definition 21 Let $\mathbb{Q} \hookrightarrow K \hookrightarrow L$ be a number field extension of degree $\ell = [L : \mathbb{Q}]$, and let $\mathcal{O}_K$ and $\mathcal{O}_L$ be the corresponding integer rings, and $\mathcal{B} = (b_1, \ldots, b_\ell)$ an integral basis of $\mathcal{O}_L$. The *size* of $x \in \mathcal{O}_L$, with x uniquely written

$$x = \sum_{i=1}^{\ell} x_i b_i, x_i \in \mathbb{Z},$$

is defined as $|x| = \log_2(\max_i(|x_i|))$, where $|x_i|$ denotes the absolute value of $x_i \in \mathbb{Z}$. The *size* of a polynomial in $\mathcal{O}_L[X; \theta]$ or, of a vector in L^n, is the maximal size of its coefficients.

An observation is that even if the inputs of **Algorithm 5** are small, and also if the output polynomial $f(X)$ is small, the size of the intermediate θ-polynomials $N(X)$ and $W(X)$ can be quite large, even though their division $f(X)$ is small. If we consider the division-free variant of the algorithm (see algorithm 7 and table of updates 4.4.1) the size of the θ-polynomials (N_i, W_i) is roughly doubled at every step.

Example 8 We consider a generalized Gabidulin code with dimension $k = 4$ and length $n = 8$ over the cyclotomic extension of $\mathbb{Q}$ with degree $m = 10$. We choose an information polynomial with size 1, and we add an error of size 2 and rank 2. After the initialization step of the algorithm, polynomials (N_0, W_0) and (N_1, W_1) are of size 12. After the last step they have size 191 (not very far from 2^4 and 2^8). In Table 4, we report timings for decoding codewords of codes of length up to 16.

5.4 Integer Gabidulin codes modulo a prime ideal

In this Section, we consider codes defined on the integer ring $\mathcal{O}_L$. We define their reduction modulo an ideal of $\mathcal{O}_L$, and study the case of Gabidulin codes.

Definition 22 Let $\mathbb{Q} \hookrightarrow K \hookrightarrow L$ be a number field, $\mathcal{C} \subset \mathcal{O}_L^n$ be an integral code and $\mathfrak{P}$ be an ideal of $\mathcal{O}_L$. The reduction of the code modulo $\mathfrak{P}$ is

$$\overline{\mathcal{C}} = \{(\overline{c}_1, \ldots, \overline{c}_n) : (c_1, \ldots, c_n) \in \mathcal{C}\}$$

where $\overline{x}$ denotes the reduction of x modulo $\mathfrak{P}$.

We want to study a Gabidulin code modulo $\mathfrak{P}$. Under conditions given in the following Theorem, this code is well defined and is a Gabidulin code over a finite field.

Theorem 13 *Let $\mathbb{Q} \hookrightarrow K \hookrightarrow L$ be a number field extension and let $\mathbb{Z}$, $\mathcal{O}_K$ and $\mathcal{O}_L$ be the associated integer rings. Let θ be a generator of the Galois group $\mathrm{Aut}_K(L)$. Let $\mathbf{g} = (g_1, \ldots, g_n) \in \mathcal{O}_L^n$ be a support of integral elements, and $\mathrm{Gab}_{\theta,k}(\mathbf{g})$ be a generalized Gabidulin code with support $\mathbf{g}$, whose associated restricted Gabidulin code is*

$$\mathcal{O}_{\mathrm{Gab}\theta,k}(\mathbf{g}) = \{(f\{g_1\}, \ldots, f\{g_n\}) : f \in \mathcal{O}_L[X; \theta], \deg(f) < k\}.$$

Finally, let $\mathfrak{P}$ be a prime ideal of $\mathcal{O}_L$, and let $\mathfrak{p} \subset \mathcal{O}_K$ denote the ideal below $\mathfrak{P}$. If the following conditions hold

1. $\theta(\mathfrak{P}) = \mathfrak{P}$,
2. $\overline{g_1}, \ldots, \overline{g_n}$ are $\mathcal{O}_K/\mathfrak{p}$-linearly independent,

then the following code

$$\overline{\mathcal{G}} = \left\{(\overline{f}\{\overline{g_1}\}, \ldots, \overline{f}\{\overline{g_n}\}) : \overline{f} \in (\mathcal{O}_L/\mathfrak{P})[X; \overline{\theta}], \deg(\overline{f}) < k\right\},$$

where $\overline{\theta}$ is defined in Eq. 14, is a classical Gabidulin code $\mathrm{Gab}_{\overline{\theta},k}(\overline{\mathbf{g}})$ defined using the extension of finite fields $\mathcal{O}_K/\mathfrak{p} \hookrightarrow \mathcal{O}_L/\mathfrak{P}$, and the automorphism $\overline{\theta}$. Furthermore, any $\overline{c} \in \overline{\mathcal{G}}$ is the reduction modulo $\mathfrak{P}$ of a codeword $c \in \mathcal{O}_{\mathrm{Gab}\theta,k}(\mathbf{g})$.

Proof Condition 2 makes $\overline{\mathbf{g}}$ a valid support of linearly elements in $\mathcal{O}_L/\mathfrak{P}$. Then $\overline{\theta}$ is a generator element of the Galois Group of the extension of finite fields $\mathcal{O}_K/\mathfrak{p} \hookrightarrow \mathcal{O}_L/\mathfrak{P}$, from Proposition 13. Actually $\overline{\theta}$ is a power of the Frobenius automorphism $x \mapsto x^q$, with $q = |\mathcal{O}_K/\mathfrak{p}|$. $\square$

Remark 9 To get $\overline{g_1}, \ldots, \overline{g_n}$ linearly independent over $\mathcal{O}_K/\mathfrak{p}$, we need $n \leqslant [\mathcal{O}_L/\mathfrak{P} : \mathcal{O}_K/\mathfrak{p}] = f$. Usually, codes are designed with $n = m$, or with n close to m. Thus, inert primes (for which $f = m$) are of a particular interest.

5.5 Decoding using a prime ideal

We exhibits a simple link between decoding a generalized Gabidulin code and decoding its reduction modulo a prime. First we provide a Lemma.

Lemma 1 *Let $\mathbf{e} = (e_1, \ldots, e_n) \in \mathcal{O}_L$ be a vector of K-rank t. Then the $\mathcal{O}_K/\mathfrak{p}$-rank of $\overline{\mathbf{e}} = \mathbf{e} \pmod{\mathfrak{P}}$ is at most t.*

Proof There exists a monic θ-polynomial E of degree t which vanishes on the e_i's. Therefore $\overline{E}\{\overline{e_i}\} = \overline{E\{e_i\}} = 0$ and $\overline{E}$ is non zero, since E is monic. From our definition of rank metric with annihilator polynomials, this implies that $\overline{\mathbf{e}}$ has rank less than or equal to t. $\square$

Theorem 14 *Let $\mathcal{G}$ be an restricted generalized Gabidulin code, $\mathfrak{p}$ be an inert prime of $\mathcal{O}_K$, such that $\overline{\mathcal{G}}$ is a generalized Gabidulin code. Suppose that $N, W \in \mathcal{O}_L[X; \theta]$ is solution to $\mathrm{LR}(n, k, t, \mathbf{g}, \mathbf{y} = (y_1, \ldots, y_n) \in \mathcal{O}_L^n)$ (see Definition 15). Let $\overline{N}, \overline{W} \in (\mathcal{O}_L/\mathfrak{P})[X; \overline{\theta}]$ be the reduction of the θ-polynomials N, W modulo $\mathfrak{P}$. Then $(\overline{N}, \overline{W})$ is a solution to $\mathrm{LR}(n, k, t, \overline{\mathbf{g}}, \overline{\mathbf{y}})$.*

Proof By hypothesis $\overline{\mathbf{g}}$ is formed with linearly independent elements, and since the conditions on the degrees of the polynomials remain unchanged by taking the polynomial modulo $\mathfrak{P}$, it is thus sufficient to check that

$$\overline{W\{y_i\}} = \overline{W}\{\overline{y_i}\}, 1 \leqslant i \leqslant n,$$
$$\overline{N\{g_i\}} = \overline{N}\{\overline{g_i}\}, 1 \leqslant i \leqslant n.$$

This is immediate from the definition of $\overline{\theta}$ in Definition 22 and from the fact that the operation of taking modulo $\mathfrak{P}$ is a ring morphism. $\square$

Suppose that one receives the vector $\mathbf{y} = f\{\mathbf{g}\} + \mathbf{e}$, where $f\{\mathbf{g}\} \in \mathcal{G}$ such that the coefficients of $f(X)$ are taken in the integer ring $\mathcal{O}_L$, and $\mathbf{e}$, such that $\mathrm{rank}\,\mathbf{e} \leqslant t \leqslant \lfloor (n - k)/2 \rfloor$, is also formed with elements of $\mathcal{O}_L$. Then

$$\overline{\mathbf{y}} = \overline{f}\{\overline{\mathbf{g}}\} + \overline{\mathbf{e}},$$

where $\overline{f}\{\overline{\mathbf{g}}\} \in \overline{\mathcal{G}}$, and $\mathrm{rank}(\overline{\mathbf{e}}) \leqslant t \leqslant \lfloor (n - k)/2 \rfloor$. In the case where $\overline{\mathcal{G}}$ is a generalized Gabidulin code over $\mathcal{O}_L/\mathfrak{P}$, then $\overline{f}(X)$ can be recovered by any algorithm solving the linear reconstruction problem like the Algorithm 5. The same is true if we consider erasures.

With this method we limit the growth of intermediate values and the bit-complexity of the decoding procedure, since all computations are completed in the finite field $\mathcal{O}_L/\mathfrak{P}$ which is isomorphic to $\mathbb{F}_{q^m}$, where $q = |\mathcal{O}_K/\mathfrak{p}|$. Recovering $\overline{f}$ requires $O(n^2)$ operations in $\mathbb{F}_{q^m}$.

Remark 10 Cyclotomic and Kummer extensions are well adapted to this purpose. Indeed, in cyclotomic extensions, there are many prime numbers of $\mathbb{Z}$ that give inert ideals. Conversely, considering Kummer extensions, many prime numbers of $\mathbb{Z}$ splits totally in the first extension, and their prime factors remain inert in the second extension.

6 Example and timings

In this Section, we provide a full example to illustrate the decoding algorithm of generalized Gabidulin codes over a number field. All steps are given. This example includes the reduction of the code modulo a prime, the correction of row and column erasures in the network coding model, the reconstruction algorithm (**Algorithm 5** and the final division). Then, we present timings for decoding a code with integer coefficients, when decoding over $\mathcal{O}_L$ (i.e. in the field L), and when decoding modulo an inert prime ideal.

6.1 A full example

In this example, we consider the extension

$$K = \mathbb{Q} \hookrightarrow L = K[\alpha] = \mathbb{Q}[Y]/(1 + Y + \cdots + Y^6)$$

provided with the K-automorphism θ defined by $\theta : \alpha \mapsto \alpha^3$. The family $(1, \alpha, \ldots, \alpha^5)$ is a K-basis of L.

We consider a code of length $n = 6$ and dimension $k = 2$. The information word has the form $f(X) = f_0 + f_1 X, f_i \in L$, where the f_i's have the form $f_i = f_{i,0} + \cdots + f_{i,5} \alpha^5$. Furthermore, suppose that only small message are encoded, i.e. $f_{i,j} \in \{0, 1\}$, and that this fact is known at the receiving end.

Since the coefficients of the information polynomial are small and belong to the integer ring of L, we can reduce the code modulo an inert prime ideal such as $3\mathcal{O}_L$. Moreover, the possibles values $\{0, 1\}$ of the coefficients $f_{i,j}$ are distinct modulo $3\mathcal{O}_L$, thus knowing f modulo $3\mathcal{O}_L$ enables to know f in L.

This example takes place in the network coding erasure model. The support $\mathbf{g} = (g_1, \ldots, g_6)$ of the code and the received word $\mathbf{y} = (y_1, \ldots, y_6)$ are the following :

$$\begin{aligned}
g_1 &= 1, & y_1 &= \alpha^5 + \alpha^3 - \alpha^2 + 2\alpha + 2 \\
g_2 &= \alpha, & y_2 &= \alpha^5 - \alpha^4 + \alpha^3 + \alpha^2 - 1 \\
g_3 &= \alpha^2, & y_3 &= -2\alpha^5 + 4\alpha^4 + \alpha^2 - 2\alpha \\
g_4 &= \alpha^3, & y_4 &= -\alpha^5 + 2\alpha^4 + \alpha^3 - \alpha^2 + 3 \\
g_5 &= \alpha^4, & y_5 &= -2\alpha^5 - 2\alpha^2 \\
g_6 &= \alpha^5, & y_6 &= -\alpha^5 - \alpha^4 + \alpha^3 - 2\alpha^2 - \alpha + 2
\end{aligned}$$

and the receiver also knows the following matrices :

$$\mathbf{A_r} = \begin{pmatrix} 1 \\ -1 \\ 0 \\ 1 \\ 1 \\ -1 \end{pmatrix} \text{ and } \mathbf{B_c} = \begin{pmatrix} 1 & 0 & -1 & 0 & 0 & 1 \end{pmatrix}$$

that describe erasures in the network coding model.

We first reduce the g_i's, the y_i's and the matrices A_r and B_c modulo $3\mathcal{O}_L$. The support of the code and the received word become the following:

$$\begin{aligned}
g_1 &= 1, & y_1 &= \overline{\alpha}^5 + \overline{\alpha}^3 + 2\overline{\alpha}^2 + 2\overline{\alpha} + 2 \\
g_2 &= \overline{\alpha}, & y_2 &= \overline{\alpha}^5 + 2\overline{\alpha}^4 + \overline{\alpha}^3 + \overline{\alpha}^2 + 2 \\
g_3 &= \overline{\alpha}^2, & y_3 &= \overline{\alpha}^5 + \overline{\alpha}^4 + \overline{\alpha}^2 + \overline{\alpha} \\
g_4 &= \overline{\alpha}^3, & y_4 &= 2\overline{\alpha}^5 + 2\overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha}^2 \\
g_5 &= \overline{\alpha}^4, & y_5 &= \overline{\alpha}^5 + \overline{\alpha}^2 \\
g_6 &= \overline{\alpha}^5, & y_6 &= 2\overline{\alpha}^5 + 2\overline{\alpha}^4 + \overline{\alpha}^3 + \overline{\alpha}^2 + 2\overline{\alpha} + 2
\end{aligned}$$

Then, we correct column erasures. We notice that the column operations $C_1 \leftarrow C_1 - C_6$ and $C_3 \leftarrow C_3 + C_6$ give a column reduced echelon form (but with pivots on the right).

We do the same operation on $\mathbf{g}$ and $\mathbf{y}$ and remove their last component (which contains all erasures) :

$$g_1 = 2\overline{\alpha}^5 + 1, \; y_1 = 2\overline{\alpha}^5 + \overline{\alpha}^4 + \overline{\alpha}^2$$
$$g_2 = \overline{\alpha}, \qquad\; y_2 = \overline{\alpha}^5 + 2\overline{\alpha}^4 + \overline{\alpha}^3 + \overline{\alpha}^2 + 2$$
$$g_3 = \overline{\alpha}^5 + \overline{\alpha}^2, \; y_3 = \overline{\alpha}^3 + 2\overline{\alpha}^2 + 2$$
$$g_4 = \overline{\alpha}^3, \qquad y_4 = 2\overline{\alpha}^5 + 2\overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha}^2$$
$$g_5 = \overline{\alpha}^4, \qquad y_5 = \overline{\alpha}^5 + \overline{\alpha}^2$$

Then, we correct row erasures. The column matrix $\mathbf{A_r}$ modulo $3\mathcal{O}_L$ correspond to the element $2\overline{\alpha}^5 + \overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha} + 1 \in \mathcal{O}_L$. We compute the annihilator polynomial $\mathcal{V}_r(X)$ of the columns of $\mathbf{A_r}$ mod $3\mathcal{O}_L$:

$$\mathcal{V}_r(X) = (\overline{\alpha}^5 + \overline{\alpha}^2)X^0 + (1)X^1$$

and evaluate this polynomial over the y_i's. We are now looking for $F(X) = \mathcal{V}_r(X) \cdot f(X)$ instead of $f(X)$. The evaluations z_i, $i = 1, \ldots, 5$ of $F(X)$ are:

$$g_1 = 2\overline{\alpha}^5 + 1, \; z_1 = 2\overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^2 + 1$$
$$g_2 = \overline{\alpha}, \qquad\; z_2 = 2\overline{\alpha}^5 + \overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha}^2 + 2\overline{\alpha} + 1$$
$$g_3 = \overline{\alpha}^5 + \overline{\alpha}^2, \; z_3 = \overline{\alpha}^5 + \overline{\alpha}^3 + \overline{\alpha}^2 + 2\overline{\alpha} + 2$$
$$g_4 = \overline{\alpha}^3, \qquad z_4 = 2\overline{\alpha}^5 + \overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha}^2 + 2\overline{\alpha}$$
$$g_5 = \overline{\alpha}^4, \qquad z_5 = 2\overline{\alpha}^5 + 2\overline{\alpha}^2 + 1$$

We now have to solve the reconstruction problem to get N and W. We now have a code of dimension 3 and length 5, no more erasures, and rank errors only, and we can apply our Welch-Berlekamp algorithm. The initialization will consider the first three evaluations and there will be 2 iterations. We use the division-free variant for this example.

At initialization, we compute the following polynomials.

$$N_0(X) = (\overline{\alpha}^5 + \overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha}^2 + 2\overline{\alpha} + 2)X^0 + (\overline{\alpha}^5 + 2\overline{\alpha}^2 + 2\overline{\alpha} + 2)X^1$$
$$\qquad\quad + (\overline{\alpha}^4 + 2\overline{\alpha} + 1)X^2 + (1)X^3$$
$$W_0(X) = 0$$
$$N_1(X) = (2\overline{\alpha}^5 + \overline{\alpha}^4 + \overline{\alpha}^2 + 2\overline{\alpha})X^0 + (2\overline{\alpha} + 2)X^1$$
$$\qquad\quad + (2\overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^3 + 2\overline{\alpha} + 1)X^2$$
$$W_1(X) = (1)X^0$$

We also initialize the discrepancies:

$$\mathbf{u}_0 = (0, 0, 0, 2\overline{\alpha} + 2, \overline{\alpha}^5 + \overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha}^2 + \overline{\alpha} + 2)$$
$$\mathbf{u}_1 = (0, 0, 0, \overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^3 + \overline{\alpha}^2 + 2\overline{\alpha} + 2, 2\overline{\alpha}^5 + \overline{\alpha}^3)$$

We begin the first iteration by extracting discrepancies:

$$u_{0,4} = N_0(g_4) - W_0(z_4)$$
$$\qquad = 2\overline{\alpha} + 2$$
$$u_{1,4} = N_1(g_4) - W_1(z_4)$$
$$\qquad = \overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^3 + \overline{\alpha}^2 + 2\overline{\alpha} + 2$$

Then we update the polynomials by the following formulas:

$$N_0'(X) = \left(X^1 - \frac{\theta(u_1)}{u_1}X^0\right) \times N_1(X)$$
$$W_0'(X) = \left(X^1 - \frac{\theta(u_1)}{u_1}X^0\right) \times W_1(X)$$
$$N_1'(X) = N_0 - \frac{u_0}{u_1}N_1(X)$$
$$W_1'(X) = W_0 - \frac{u_0}{u_1}W_1(X).$$

We switch them to get:

$$
\begin{aligned}
N_0(X) = N_1'(X) =\ & (2\overline{\alpha}^5 + 2\overline{\alpha}^4 + \overline{\alpha} + 1)X^0 + (2\overline{\alpha}^5 + \overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha}^2)X^1 \\
& + (\overline{\alpha}^5 + 2\overline{\alpha}^4 + \overline{\alpha}^3 + \overline{\alpha}^2 + 2\overline{\alpha})X^2 \\
& + (2\overline{\alpha}^5 + 2\overline{\alpha}^3 + 2\overline{\alpha}^2 + 2\overline{\alpha} + 1)X^3 \\
W_0(X) = W_1'(X) =\ & (2\overline{\alpha}^5 + \overline{\alpha}^4 + 2\overline{\alpha}^3 + 2\overline{\alpha} + 1)X^0 + (1)X^1 \\
N_1(X) = N_0'(X) =\ & (2\overline{\alpha}^3 + \overline{\alpha}^2 + 2\overline{\alpha} + 2)X^0 + (\overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^3 + 2\overline{\alpha}^2 + \overline{\alpha})X^1 \\
& + (2\overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha})X^2 + (1)X^3 \\
W_1(X) = W_0'(X) =\ & (2\overline{\alpha}^5 + 2\overline{\alpha} + 1)X^0
\end{aligned}
$$

We also update and switch discrepancies, to get:

$$
\begin{aligned}
\mathbf{u}_0 &= (0,\, 0,\, 0,\, 0,\, 2\overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^3 + 2\overline{\alpha} + 1) \\
\mathbf{u}_1 &= (0,\, 0,\, 0,\, 0,\, 2\overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^3 + \overline{\alpha}^2 + 2\overline{\alpha} + 1)
\end{aligned}
$$

During the second iteration, we get:

$$
\begin{aligned}
u_{0,5} &= N_0(g_4) - W_0(g_5) \\
&= 2\overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^3 + 2\overline{\alpha} + 1 \\
u_{1,5} &= N_1(g_4) - W_1(z_5) \\
&= 2\overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^3 + \overline{\alpha}^2 + 2\overline{\alpha} + 1
\end{aligned}
$$

and

$$
\begin{aligned}
N_0(X) = N_1'(X) =\ & (2\overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha}^2 + \overline{\alpha} + 1)X^0 + (2\overline{\alpha}^5 + \overline{\alpha}^4 + \overline{\alpha}^2 + 2)X^1 \\
& + (\overline{\alpha}^4 + \overline{\alpha}^3 + \overline{\alpha}^2 + 1)X^2 + (2\overline{\alpha}^5)X^3 \\
& + (1)X^4 \\
W_0(X) = W_1'(X) =\ & (2\overline{\alpha}^4 + 2\overline{\alpha}^3 + 2\overline{\alpha}^2 + 2\overline{\alpha} + 1)X^0 + (2\overline{\alpha}^3 + 2\overline{\alpha} + 1)X^1 \\
N_1(X) = N_0'(X) =\ & (2\overline{\alpha}^5 + 2\overline{\alpha}^3)X^0 + (\overline{\alpha}^5 + \overline{\alpha}^3 + \overline{\alpha}^2 + \overline{\alpha} + 2)X^1 \\
& + (\overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^3 + 2\overline{\alpha}^2 + \overline{\alpha} + 1)X^2 + (\overline{\alpha}^3)X^3 \\
W_1(X) = W_0'(X) =\ & (\overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha}^2 + \overline{\alpha} + 1)X^0 + (1)X^1
\end{aligned}
$$

Updating and switching discrepancies, we get

$$
\begin{aligned}
\mathbf{u}_0 &= (0,\, 0,\, 0,\, 0,\, 0) \\
\mathbf{u}_1 &= (0,\, 0,\, 0,\, 0,\, 0)
\end{aligned}
$$

which satisfy the interpolation condition. Then, the polynomials

$$
\begin{aligned}
N(X) = N_1(X) =\ & (2\overline{\alpha}^5 + 2\overline{\alpha}^3)X^0 + (\overline{\alpha}^5 + \overline{\alpha}^3 + \overline{\alpha}^2 + \overline{\alpha} + 2)X^1 \\
& + (\overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^3 + 2\overline{\alpha}^2 + \overline{\alpha} + 1)X^2 + (\overline{\alpha}^3)X^3 \\
W(X) = W_1(X) =\ & (\overline{\alpha}^4 + \overline{\alpha}^3 + 2\overline{\alpha}^2 + \overline{\alpha} + 1)X^0 + (1)X^1
\end{aligned}
$$

are solution of the reconstruction problem. We divide them to get

$$
F(X) = (\overline{\alpha}^4 + 1)X^0 + (2\overline{\alpha}^5 + 2\overline{\alpha}^4 + 2\overline{\alpha}^2 + 2\overline{\alpha})X^1 + (\overline{\alpha})X^2,
$$

then, we divide it by $\mathcal{V}_r(X)$ to recover

$$
f(X) = (\overline{\alpha}^2)X^0 + (\overline{\alpha}^5)X^1.
$$

Thus, since the possible $f_{i,j}$ are 0 or 1 modulo $\mathfrak{P}$, we have recovered the information polynomial $f(X) = \alpha^2 X^0 + \alpha^5 X^1$.

length n of the code	4	6	8	10	12	14	16
degree $[\mathbb{Q}[\alpha] : \mathbb{Q}]$	4	6	10	10	12	16	16
prime ideal $\mathfrak{P}$	$2\mathcal{O}_L$	$3\mathcal{O}_L$	$2\mathcal{O}_L$	$2\mathcal{O}_L$	$2\mathcal{O}_L$	$3\mathcal{O}_L$	$3\mathcal{O}_L$

Table 3 Cyclotomic extensions used in our timings

$n\backslash k$	2	4	6	8	10	12	14	16
4	0.07	0.08						
6	0.23	0.24	0.21					
8	0.84	0.85	0.87	0.85				
10	1.78	1.91	2.11	2.32	2.51			
12	10.59	11.00	12.52	15.15	17.72	20.32		
14	215	196	202	242	292	345	398	
16	1522	1320	1405	1722	2061	2503	2887	3223

Table 4 Timings (in seconds) over the number field L

$n\backslash k$	2	4	6	8	10	12	14	16
4	0.10	0.09						
6	0.33	0.31	0.25					
8	0.83	0.83	0.76	0.62				
10	1.33	1.37	1.32	1.19	1.00			
12	2.33	2.46	2.46	2.31	2.06	1.78		
14	5.13	5.41	5.42	5.20	4.78	4.25	3.72	
16	6.69	7.20	7.31	7.17	6.74	6.20	5.53	4.96

Table 5 Timings (in seconds) using the residue field $\mathcal{O}_L/\mathfrak{P}$

6.2 Timings

For timings measurements, our generalized Gabidulin codes are constructed over a cyclotomic extension $\mathbb{Q} \hookrightarrow L = \mathbb{Q}[\alpha] = \mathbb{Q}[Y]/(1 + Y + \cdots + Y^{m-1})$, for small values of m, see Table 3. The information words are on the form $f(X) = \sum_{i=0}^{k-1} f_i X^i$. The error is constructed by making the product of a vector $(e_1, \ldots, e_t)$ by a matrix of t rows and n columns over K. In order to have small coefficients, the f_i's and the e_i's are on the form $\sum_{j=1}^{m} x_j \alpha^j$ with $x_j \in \{0; 1\}$ and the expanded coefficients are chosen in $\{-1; 0; 1\}$.

First, computations are done in $\mathcal{O}_L$, since all coefficients are integral. Then, the received words are reduced modulo an inert prime ideal $\mathfrak{P}$. These reduced received words are decoded over $\mathcal{O}_L/\mathfrak{P}$. This ideal is generated by the smallest prime number of $\mathbb{Z}$ inert in $\mathcal{O}_L$. The time required for computation are respectively presented in **Tables 4 and 5**.

The algorithm has been written in Magma V2.20-9. The machine has 24 processors intel xeon X5690, 96 gigas of RAM, is 3.47gHz clocked, and has distribution ubuntu 14. Time computation are obtained with the Cputime function. It corresponds to the time required to 50 runs of decoding algorithm.

7 Conclusion

Given any cyclic Galois extension $K \hookrightarrow L$ provided with an automorphism θ generating the Galois group $\mathrm{Aut}_K(L)$, we can design generalized Gabidulin codes. Cyclotomic,

Kummer or Artin-Schreier extensions are examples of extensions that fulfill the condition which enable to design the codes.

We also have provided various useful definitions of the rank metric. These generalized Gabidulin codes have the same properties as their analogues in finite fields, namely they are also MRD, and can be decoded with an adaptation of the Welch-Berlekamp algorithm with quadratic complexity in terms of operations in L. Such a code, with parameters $[n, k, d]_r$, enables to correct up to s_c column erasures, s_r row erasures and an error of rank t if $s_c + s_r + 2t \leqslant n - k$ in both line erasure and network coding erasure models.

Of course, over an infinite field, one obstacle is the growth of intermediate coefficients. We can circumvent this problem computing modulo an inert prime ideal of $\mathbb{Z}$, chosen large enough when the size of the message or the error is known, by observing that a Generalized Gabidulin code modulo a prime ideal is a classical Gabidulin code over a finite field.

References

[Ber68]   E. R. Berlekamp. *Algebraic Coding Theory*. McGraw-Hill, 1968.

[BGU07] Delphine Boucher, Willi Geiselmann, and Felix Ulmer. Skew-cyclic codes. *Applicable Algebra in Engineering, Communication and Computing*, 18(4):379–389, 2007.

[BM85]   M. Blaum and R. J. McEliece. Coding protection for magnetic tapes: A generalization of the Patel-Hong code. *Information Theory, IEEE Transactions on*, 31(5):690–693, 1985.

[BU14]   Delphine Boucher and Felix Ulmer. Linear codes using skew polynomials with automorphisms and derivations. *Designs, Codes and Cryptography*, 70(3):405–431, 2014.

[BW86]   Elwyn R Berlekamp and Lloyd R Welch. Error correction for algebraic block codes, December 30 1986. US Patent 4,633,470.

[Coh93]   Henri Cohen. *A course in computational algebraic number theory*. Graduate text in mathematics. Springer, 1993.

[Del78]   Philppe Delsarte. Bilinear forms over a finite field, with applications to coding theory. *Journal of Combinatorial Theory, Series A*, 25(3):226–241, 1978.

[FL06]   Cédric Faure and Pierre Loidreau. A new public-key cryptosystem based on the problem of reconstructing px-polynomials. In *Proceedings of the 2005 International Conference on Coding and Cryptography*, WCC'05, pages 304–315, Berlin, Heidelberg, 2006. Springer-Verlag.

[Gab85]   Ernest Mukhamedovich Gabidulin. Theory of codes with maximum rank distance. *Problemy Peredachi Informatsii*, 21(1):3–16, 1985.

[GPT91]   E. M. Gabidulin, A. V. Paramonov, and O. V. Tretjakov. Ideals over a noncommutative ring and their application in cryptology. In D. W. Davies, editor, *Advances in Cryptology - EUROCRYPT'91*, volume 547 of *Lecture Notes in Computer Science*, pages 482–489. Springer, 1991.

[GS92]   Peter Gemmell and Madhu Sudan. Highly resilient correctors for polynomials. *Information processing letters*, 43(4):169–174, 1992.

[KK08]   Ralf Koetter and Frank R Kschischang. Coding for errors and erasures in random network coding. *IEEE Transactions on Information Theory*, 54(8):3579–3591, 2008.

[LN97]   Rudolf Lidl and Harald Niederreiter. *Finite fields*. Cambridge university press, 1997.

[Loi06]   Pierre Loidreau. A Welch–Berlekamp like algorithm for decoding Gabidulin codes. In Øyvind Ytrehus, editor, *Coding and Cryptography: International Workshop, WCC 2005, Bergen, Norway, March 14-18, 2005. Revised Selected Papers*, pages 36–45. Springer, 2006.

[LSS14]   Wenhui Li, Vladimir Sidorenko, and Danilo Silva. On transform-domain error and erasure correction by Gabidulin codes. *Designs, Codes and Cryptography*, 73(2):571–586, 2014.

[Neu99]   Jürgen Neukirch. *Algebraic number theory*. Springer, 1999.

[Ore33a]   Oystein Ore. On a special class of polynomials. *Transactions of the American Mathematical Society*, 35(3):559–584, 1933.

[Ore33b] Oystein Ore. Theory of non-commutative polynomials. *Annals of mathematics*, 34(3):480–508, 1933.

[Ore34]   Oystein Ore. Contributions to the theory of finite fields. *Transactions of the American Mathematical Society*, 36(2):243–274, 1934.

[Rot91]   Ron M. Roth. Maximum-rank array codes and their application to crisscross error correction. *Information Theory, IEEE Transactions on*, 37(2):328–336, 1991.

[Sam71]   Pierre Samuel. *Théorie algébrique des nombres*. Hermann, 1971.