Reed-Solomon-Gabidulin Codes
Xavier Caruso, Amaury Durand

To cite this version:
Xavier Caruso, Amaury Durand. Reed-Solomon-Gabidulin Codes. 2018. <hal-01949409v2>
Reed–Solomon–Gabidulin Codes

Xavier Caruso* and Amaury Durand†

January 14, 2019

Abstract

We introduce Reed-Solomon-Gabidulin codes which is, at the same time, an extension to Reed-Solomon codes on the one hand and Gabidulin codes on the other hand. We prove that our codes have good properties with respect to the minimal distance and design an efficient decoding algorithm.

Important disclaimer

After we made this article available on HAL and arXiv, we received an email from Umberto Martinez-Penas, in which he kindly explained to us that the results obtained in the present article were already discovered (and partly published) recently [5, 6]; our notion of Reed–Solomon–Gabidulin codes is actually a special case of the notion of Linearized Reed-Solomon codes introduced there.

Nevertheless, our exposition differs a bit from that of loc. cit, so we think that our article still has some interest. Combined with the results of [3], our version of the decoding algorithm has sub-quadratic complexity; this was left as an open question in [6].

Introduction

Reed–Solomon codes form a well-known class of error detection and correction codes which have very interesting properties (optimal minimal distance, efficient decoding algorithms). They were introduced in 1960 by Reed and Solomon and are nowadays widely used in everyday life. About twenty years later, Delsarte [4], Gabidulin [7] and Roth [13]—independently—imagined an analogue of Reed–Solomon codes in the context of the rank distance, which is finer than the standard Hamming distance and well suited for some applications (e.g. network coding). These codes are nowadays called Gabidulin codes. Their construction is based on the concept of linearized polynomials over the finite fields. More recently several authors generalized and optimized Gabidulin codes. In 2013, in her thesis [14] and subsequent papers, Wachter-Zeh proposed an efficient implementation of operations with linearized polynomials, together with an equivalent of Gao’s decoding algorithm.

In 2009, Boucher, Geiselmann and Ulmer [1] introduced analogues of BCH codes in the Gabidulin’s context of linearized polynomials (cf also [2]). It worths

*CNRS, Institut Mathématique de Bordeaux, équipe LFANT
†Université de Franche-Comté
mentionning that they use Ore polynomials (introduced by Ore in 1933 in [10])
in place of linearized polynomials. Although the two approaches are equivalent
in the case of finite fields, it turns out that Ore polynomials are more general
objects which continue to make sense in a large variety of settings. Taking
advantage of this new point of view, Robert proposed in his thesis [12] an
extension of Gabidulin’s code to the caracteristic zero, in which basically finite
fields are replaced by number fields.

Another advantage of Boucher, Geiselmann and Ulmer’s approach is that it
allows longer codes: while the length of a Gabidulin code is necessarily bounded
from above by the degree of the finite field we are working with, this bound can
be generally overpassed in Boucher, Geiselmann and Ulmer’s construction. On
the other hand, no efficient decoding algorithm is known.

**Contribution of the article.** In the present paper, we introduce and study
a new generalization of Gabidulin codes, which combines all the benefits of
previous constructions. Precisely, we shall show that:
(1) as for Gabidulin codes, our codes are MDS (Maximal Distance Separable),
(2) as in Boucher, Geiselmann and Ulmer’s work, long codes are permitted,
(3) as in Wachter-Zeh’s work, there exists an efficient decoding algorithm.

Besides, the setting we consider includes the case of finite fields (as in Gabidulin’s
initial definition) and number fields (as in Robert’s generalization) but it is even
more general. For example, our construction allows the base field to be the field
of rational fractions in the variable \( t \) over a finite field equipped with its canoni-
cal derivation \( \frac{d}{dt} \).

Moreover it turns out that, for a special choice of parameters, our codes
extend classical Reed–Solomon codes. For this reason, we have decided to call
them **Reed–Solomon–Gabidulin (RSG) codes**.

**Organization of the article.** This paper is divided in two sections. The first
one is devoted to introduce and develop the necessary background on Ore polyno-
mials and related notions. We will study particularly the notion of evaluation
morphisms which is the main ingredient we will need for defining GRS codes.
In the second section, we introduce GRS codes and state their main properties
(cf (1), (2), (3) above). For the sake of brievity, proofs are omitted though
intermediate steps are often isolated.

## 1 Ore polynomials

Throughout this article, we use the following notation: \( K \) is a field, \( \theta : K \to K \)
be a ring homomorphism and \( \partial : K \to K \) be a \( \theta \)-derivation, i.e. an additive
mapping such that \( \partial(ab) = \theta(a)\partial(b) + \partial(a)b \) for all \( a, b \in K \).

We shall denote by \( F \) the subfield of \( K \) consisting of elements \( a \) such that
\( \theta(a) = a \) and \( \partial(a) = 0 \). **We will always assume that the extension \( K/F \)
is finite** and will denote by \( r \) its degree. Our assumption implies in particular
that \( \theta \) has finite order and thus is bijective.

\(^{1}\)Be careful at not making the confusion with GRS codes, which stands for **Generalized
Reed–Solomon codes**.
Definition 1.1 (Ore polynomial ring). The ring of Ore polynomials $K[X; \theta, \partial]$ is the ring whose elements are polynomials in $X$ over $A$ endowed with the usual addition and with the multiplication defined by the rule:

$$X \times a = \theta(a)X + \partial(a), \; \forall a \in A.$$ 

Example 1.2. Throughout this article, we will illustrate our constructions with the two following examples:

(1) (This setting is the one in which Gabidulin codes were first defined by Gabidulin in [7], with a slightly different vocabulary.) Let $p$ be a prime number, $q$ be a power of $p$ and $r$ be a positive integer. We let $F_{q^r}$ denote a finite field with cardinality $q^m$. We endow it with the Frobenius $\text{Frob}_q : x \mapsto x^q$. The first Ore ring we will be interested in is $F_{q^r}[X; \text{Frob}_q, 0]$. In this setting, the subfield $F$ of $K = F_{q^r}$ we have introduced is $F_q$. The degree of the extension $K/F$ is then $r$.

(1’) More generally, one can pick an arbitrary field $K$, endow it with a finite order automorphism $\theta$ and consider the Ore ring $K[X, \theta, 0]$. Beyond the case of finite fields, natural examples are cyclotomic extensions of $\mathbb{Q}$ or Kummer extensions. This case was addressed in Robert’s thesis [12].

(2) Let $\kappa$ be a field of characteristic $p$. We consider the field $K = \kappa(t)$ and endow it with the natural derivation $\frac{d}{dt}$. We can then form the Ore ring $\kappa(t)[X, \text{id}, \frac{d}{dt}]$. Here the subfield $F$ of $K$ is $\kappa(t^p)$ and the degree of the extension $K/F$ is then $p$.

The notion of degree extends verbatim to Ore polynomials: if $P = \sum a_i X^i$ is an Ore polynomial, its degree is the largest integer $i$ for which $a_i \neq 0$. Besides, one can prove the existence of a right Euclidean division for Ore polynomials: if $A, B \in K[X; \theta, \partial]$ with $B \neq 0$, there exist unique $Q, R \in K[X; \theta, \partial]$ with $A = QB + R$ and $\deg R < \deg B$. This has the usual consequences: the non-commutative ring $K[X; \theta, \partial]$ is left-principal, right $\text{gcd}$s and left $\text{lcm}$s are well defined and can be computed by Euclidean algorithm. Similarly, left Euclidean divisions, left $\text{gcd}$s and right $\text{lcm}$s do exist (since our general assumptions imply that $\theta$ is bijective).

Notation: In what follows, we will denote by $A \% B$ the remainder in the right division of $A$ by $B$.

The centre.

Recall that the centre of a noncommutative ring $A$ is by definition the subset of $A$ consisting of elements $x$ such that $xy = yx$ for all $y \in A$. We observe in particular that the centre of $A$ is a commutative subring of $A$. In the case of Ore polynomials, the centre can actually be computed precisely. In what follows, we will not need a complete description but only the general structure of the centre as given by the next proposition.

Proposition 1.3. There exists a central Ore polynomial $Z(X) \in K[X; \theta, \partial]$ of degree $r$ such that the centre of $K[X; \theta, \partial]$ is $F[Z(X)]$, i.e. the subset of Ore polynomials that can be written as a polynomial in $Z(X)$ with coefficient in $F$. 

3
We observe that the equality:
\[ a_0 + a_1 Z(X) + \cdots + a_d Z(X)^d = b_0 + b_1 Z(X) + \cdots + a_e Z(X)^e \]
implies readily that \( d = e \) (compare the degrees) and \( a_i = b_i \) for all \( i \). As a consequence the centre \( F[Z(X)] \) is an actual (commutative) ring of univariate polynomials with coefficients in \( F \).

On the other hand, we draw the attention of the reader to the fact that the properties of Proposition 1.3 do not determine \( Z(X) \) uniquely but only up to an additive constant in \( F \).

Example 1.4. We continue Example 1.2. In the settings (1) and (1'), it is easily seen that the centre of \( K[X; \theta, 0] \) is \( F[X] \). In the setting (2), the centre of \( \kappa(t)[X; \text{id}, \frac{dt}{t}] \) (where \( \kappa \) is a field of characteristic \( p \)) is \( \kappa(t^p)[X^p] \).

Pseudo-linear morphisms.

Another important notion is that of pseudo-linear morphisms. It is defined as follows:

Definition 1.5 (Pseudo-linear morphism). Let \( M \) and \( N \) be two vector spaces over \( K \). A pseudo-linear morphism \( u : M \to N \) is a map verifying \( u(ax) = \theta(\alpha)u(x) + \partial(\alpha)x \) for all \( \alpha \in K \) and \( x \in M \).

We observe that any pseudo-linear morphism is \textit{a fortiori} \( F \)-linear (where \( F \) is defined at the beginning of this section).

Pseudo-linear morphisms are relevant in the context of Ore polynomials because the Ore multiplication reflects the composition rule of pseudo-linear morphisms. More precisely, given a pseudo-linear endomorphism \( u : M \to M \) and an Ore polynomial \( P = \sum a_i X^i \in K[X; \theta, \partial] \), one defines \( P(u) = \sum a_i u^i \). One then easily checks that \( P(u) \circ Q(u) = (PQ)(u) \) where the multiplication on the right hand size is the Ore multiplication. In other words, denoting by \( \text{End}_F(M) \) the ring of \( F \)-linear maps from \( M \) to itself, the “evaluation” mapping

\[ \text{ev}_u : K[X; \theta, \partial] \to \text{End}_F(M), \quad P(X) \mapsto P(u) \]

is a ring homomorphism for any pseudo-linear endomorphism \( u \).

The case where \( M \) is \( K \) itself deserves particular attention. Indeed, we first observe that evaluation is then closely related to Euclidean division thanks to the formula:

\[ \text{ev}_u(P)(\alpha) = a \cdot P \% (X - \frac{u(\alpha)}{a}) \quad (1) \]

which is correct for any pseudo-linear endomorphism \( u \) of \( K \), any \( P \in K[X; \theta, \partial] \) and any \( \alpha \in K \). Second, we have a complete classification of pseudo-linear endomorphisms of \( K \).

Proposition 1.6. The pseudo-linear endomorphisms of \( K \) are exactly the maps of the form \( \partial + c \theta \) with \( c \in K \).

In what follows, we will often use the notation \( \text{ev}_c \) in place of \( \text{ev}_{\partial + c \theta} \).
Main properties of the $ev_c$'s. We denote by $K_{\text{good}}$ the subset of $K$ consisting of elements $c$ for which $\partial + c\theta$ is not of the form $a \cdot \text{id}$ with $a \in F$. Except in the very particular case where $\theta = \text{id}$ and $\partial = 0$ (where $K_{\text{good}}$ is obviously empty), one can prove that there is at most one bad value of $c$, i.e. the difference between $K$ and $K_{\text{good}}$ consists at most of one element.

**Proposition 1.7.** For all $c \in K_{\text{good}}$, the ring homomorphism $ev_c$ is surjective and its kernel is a principal ideal generated by $Z(X) - N(c)$ for some element $N(c) \in F$.

**Remark 1.8.** The function $N$ defined by Proposition 1.7 above is not canonical since it depends on the choice of the constant coefficient of $Z(X)$. Two different choices lead to functions $N$ and $N'$ such that $N' = N + a$ for some constant $a \in F$.

**Definition 1.9.** Let $c_1, c_2 \in K_{\text{good}}$. We say that $c_1$ and $c_2$ are equivalent if $\ker ev_{c_1} = \ker ev_{c_2}$ or, equivalently, $N(c_1) = N(c_2)$.

Using Noether–Skolem Theorem, one can prove the following characterization:

**Lemma 1.10.** The elements $c_1$ and $c_2$ are equivalent if and only if there exists $a \in K$, $a \neq 0$ such that $c_1 a = c_2 \theta(a) + \partial(a)$.

In particular, the equivalence class of $c \in K$ is exactly the image of $x \mapsto \frac{\theta(a)}{x}$.

**Example 1.11.** Let us first focus on the settings (1) and (1') of Example 1.2. The subset $K_{\text{good}}$ is then $K \setminus \{0\}$. Moreover if we have chosen $Z(X) = X^r$ (see Example 1.4), it is not difficult to prove that the map $N$ is the norm of $K$ over $F$. In this context, the characterization of Lemma 1.10 is a classical consequence of Hilbert 90 theorem which says that an element has norm 1 if and only if it can be written $\theta(a)$ for some $a \neq 0$.

When $K = F_q$ and $\theta = \text{Frob}_q$, we have $N(c) = c^{1+q+q^2+\cdots+q^{m-1}}$. In this case, the image of $N$ is $F_q^*$ and there is exactly $q-1$ equivalence classes for the equivalence relation introduced in Definition 1.9.

In the setting (2), we have $K_{\text{good}} = K$. Moreover, with the normalization $Z(X) = X^p$, one can prove\footnote{Through the proof is not obvious.} that $N(f) = f^{d^{-1}f}$ for any $f \in k(t)$. Here, Lemma 1.10 asserts that $N(f) = N(g)$ if and only if the difference $f - g$ is a logarithmic derivative. It is easily seen that a polynomial cannot be a logarithmic derivative. Consequently the elements of $k[t]$ are pairwise nonequivalent, implying in particular that there are infinitely many equivalence classes for this relation.

### 2 Reed–Solomon–Gabidulin codes

We keep the notations of the previous section. In particular, we recall that $K_{\text{good}}$ is the subset of $K$ consisting of elements $c$ for which $\partial + c\theta$ is not of the form $a \cdot \text{id}$ with $a \in F$.\footnotemark
We set 

In particular, the map has degree $n\cdot 2$. 

Example With these parameters, we easily compute $a$. 

Thanks to Eq. (1), the mapping $\gamma_{c,g}$ can be rewritten in terms of Euclidean divisions. More precisely, for $1 \leq i \leq s$ and $1 \leq j \leq n_i$, letting: 

$$a_{i,j} = \frac{(\partial + c_i\theta)(g_{i,j})}{g_{i,j}}$$

we have $ev_{c_i}(g_{i,j}) = g_{i,j} \cdot P \% (X - a_{i,j})$. 

For any positive $k$, we let $\gamma_{k,c,g}$ denote the restriction of $\gamma_{c,g}$ to the subspace $K[X;\theta,\partial]_{\leq k}$ consisting of Ore polynomials of degree less than $k$. 

**Example 2.1.** Consider the setting (1) of Example 1.2. Let $g$ be a multiplicative generator of $F_q^*$. Its norm over $F_q$ is a multiplicative generator of $F_q^*$. By what we did in Example 1.11, the elements $c_i = g^i$ for $0 \leq i < s$ are pairwise nonequivalent as soon as $s \leq q - 1$. (Here, for simplicity, we have shifted our indices so that they start from 0 instead of 1.) Moreover (1, $g, \ldots, g^{p-1}$) is a basis of $F_q^*$ over $F_q$. One can then take $n_i = r$ for all $i$ and $g_{i,j} = g^j$ for $0 \leq j < r$. With these parameters, we easily compute $a_{i,j} = c_i \cdot \text{Frob}_{g}(g_{i,j}) \cdot g_{i,j}^{-1} = g^{i+(q-1)j}$. 

**Example 2.2.** Consider the setting (2) of Example 1.2. By Example 1.11 again, we can take any family $(c_1, \ldots, c_s)$ of pairwise distinct polynomials. Moreover a basis of $\kappa(t^p)$ over $\kappa(t)$ is obviously $(1, t, \ldots, t^{p-1})$. Therefore, we can take $n_i = p$ and $g_{i,j} = t^j$ for $0 \leq j < p$. A direct computation leads to $a_{i,j} = \frac{1}{t} + c_i$. Taking $\kappa = F_3, k = 5, c = (0, 1)$ and $g = ((1, t, t^2), (1, t, t^2))$, we find that the matrix of $\gamma_{k,c,g}$ is: 

$$
\begin{pmatrix}
1 & t & t^2 \\
0 & 1 & 2t \\
1 & t+1 & t^2+2t
\end{pmatrix}.
$$

The kernel of $\gamma_{k,c,g}$ is the principal ideal generated by the Ore polynomial: 

$$L = \text{LLCM}((X - a_{i,j})_{1 \leq i \leq m, 1 \leq j \leq n_i}).$$

The next lemma shows that the assumption we made on the $c_i$'s and $g_{i,j}$'s are directly related to the degree of $L$. 

**Lemma 2.3.** With the above notations and assumptions, the Ore polynomial $L$ has degree $n$. In particular, the map $\gamma_{n,c,g}$ is bijective.
Example 2.4. Continuing Example 2.1, the Ore polynomial $L$ defined in (4) is
\[ L = \prod_{i=1}^{s}(X^r - N(c_i)) \] where we recall that $N : F_q \to F_q$ is the norm map. (Observe that the factors $X^r - N(c_i)$ all lie in the centre of $F_q[X; \text{Frob}_q, 0]$ so that the product we have written in not ambiguous.) In particular, when $s = q - 1$, we get $L(X) = X^{r(q - 1)} - 1$.

Example 2.5. Continuing Example 2.2 and assuming further that the $c_i$’s lie in $\kappa$, we find that the polynomial $L$ defined in (4) is $L = \prod_{i=1}^{s}(X^p - c_i^q)$. In particular, if $\kappa$ is a finite field of cardinality $q$ and the $c_i$’s enumerate the elements of $\kappa$ (so that $s = q$), we have $L(X) = X^{pq} - X^p$.

Definition and first properties.

We are now ready to define Gabidulin codes in the extended framework discussed in the introduction of this section.

**Definition 2.6.** With the previous notations, the Reed–Solomon–Gabidulin (RSG for short) code $\text{RSG}_{k,c,g}$ associated to $c$ and $g$ is the image of $\gamma_{k,c,g}$.

**Remark 2.7.** From the definition, it follows that the matrix of $\gamma_{k,c,g}$ (in the canonical basis) is a generator matrix of $\text{RSG}_{k,c,g}$. The matrix (3) then provide an example of a generator matrix of a RSG code.

It is well known that the relevant distance for Gabidulin codes is not the Hamming distance but the rank distance. In the context of Gabidulin codes introduced above, we shall need another distance which is a mixture between Hamming distance and rank distance. It is defined as follows.

**Definition 2.8.** Let $x = (x_{i,j})_{1 \leq i \leq n_1, 1 \leq j \leq n_j} \in K^{n_1} \times K^{n_2} \times \cdots \times K^{n_s}$. The rank-Hamming weight of $x$ is:
\[ w_{rH}(x) = \sum_{i=1}^{s} \dim_{F}(x_{i,1}, x_{i,2}, \ldots, x_{i,n_i}). \]

Given $x, y \in K^{n_1} \times K^{n_2} \times \cdots \times K^{n_s}$, the rank-Hamming distance between $x$ and $y$ is
\[ d_{rH}(x, y) = w_{rH}(x - y). \]

**Remark 2.9.** The weight $w_{rH}$ is finer that the usual Hamming weight in the sense that, for all $x \in K^{n_1} \times \cdots \times K^{n_s}$, we have $w_{rH}(x) \leq w_{H}(x)$ if $w_{H}$ denotes the Hamming weight.

The RSG codes we have defined extend the classical notion of Gabidulin codes introduced in [7]. More precisely, the latter correspond to the case where $s = 1$, $\partial = 0$ and $K$ is a finite field. Relaxing the assumption on $K$, we obtain the generalized Gabidulin codes defined by Robert in his thesis [12]. In particular, in this case, the rank-Hamming distance is the usual rank distance.

On the other hand, when $\theta = \text{id}$ and $\partial = 0$ (that is $F = K$), the notion of RSG code is nothing but the standard notion of Reed–Solomon code and the rank-Hamming distance reduces to the usual Hamming distance.

**Proposition 2.10.** The code $\text{RSG}_{k,c,g}$ has length $n$, dimension $k$ and minimal distance $d = n - k + 1$.

**Example 2.11.** The RSG code corresponding to the generator matrix (3) has length 6, dimension 2 and minimal distance $6 - 2 + 1 = 5$. It then corrects any error of rank-Hamming weight at most 2.
Decoding Reed–Solomon–Gabidulin codes.

RSG codes can be decoded by a noncommutative extension of Gao’s algorithm [8]. This fact was already observed in the works of Wachter-Zeh and al. [14] in the special case of usual Gabidulin codes. After what we have done previously, the extension to RSG codes is not difficult.

Gao’s algorithm consists in several steps that we will present below. We suppose that we are given parameters $k$, $c$ and $g$ as above together with a codeword $c = \gamma_k, c, g(P)$ for an Ore polynomial $P$ of degree less than $k$. Let $w$ denote the ceiling of $\frac{n-k}{2}$ and let $e \in K^{n_1} \times \cdots \times K^{n_s}$ be a vector of rank-Hamming weight at most $w$. We set $m = c + e$.

**Example 2.12 (Thread example).** We shall illustrate each step of Gao’s algorithm by the following thread example. As in Example 3, we take $K = \mathbb{F}_3(t)$ (equipped with $\theta = \text{id}$ and $\partial = \frac{d}{dt}$), $k = 2$, $c = (0, t)$ and $g = ((1, t^3), (1, t, t^2))$. The generator matrix of the corresponding RSG code is the matrix (3). We will work with the following codeword:

$$c = \gamma_{k, c, g}(t^2X + 1) = \left( (1, t^2 + t, 2t^3 + t^2), (t^2 + 1, t^3 + t^2 + t, t^4 + 2t^3 + t^2) \right)$$

and the following error $e = \left( (1, t^3, 2t^3), (t + 1, 0, t^4 + t^3) \right)$ which has rank-Hamming weight 2. The corresponding received message is:

$$m = \left( (2, t^3 + t^2 + t, t^3 + t^2), (t^2 + t + 2, t^3 + t^2 + t, 2t^4 + t^2) \right).$$

**Step 0: Annihilator.** We compute the Ore polynomial $L$ defined in (4).

If a fast multiplication algorithm of Ore polynomials is available (which is notably the case when $\partial = 0$ [11, 3]), this computation can be done efficiently by a divide-and-conquer algorithm [3].

We underline that this computation is independant of the received message $m$ and then has to be done just once when the RSG code is set up.

**Example 2.13.** In our thread example, we have $L(X) = X^6 - X^3$ as shown by Example 2.5.

**Step 1: Interpolation.** We compute a Ore polynomial $\tilde{P}$ of degree less than $n$ such that $\gamma_{c, g}(P) = m$.

This can be done for example by inverting the $K$-linear map $\gamma_{n, c, g}$, which is known to be a bijection by Lemma 2.3. Alternatively, $\tilde{P}$ can be computed by solving a (noncommutative) Chinese remainder problem. This latter approach is faster when an efficient multiplication algorithm of Ore polynomials is available.

**Example 2.14.** In our thread example, we find:

$$\tilde{P} = (2t^4 + t^2)X^4 + (2t^4 + t^3 + 2t)X^3 + (2t^4 + t^3 + 2t^2)X^2 + (t^3 + t^2 + 2t)X + 2.$$
Step 2: Partial rgcd. We compute a relation of the form \( U\tilde{P} + VL = R \) for Ore polynomials \( U, V \) and \( R \) with \( \deg U \leq w \) and \( \deg R < w + k \). This relation can be computed by applying the extended Euclidean algorithm with the input \( (\tilde{P}, L) \) and stopping it the first time the remainder \( R \) has degree less than \( w + k \).

Remark 2.16. Using the theory of resultants and subresultants [9], one can carry out this computation by controlling the degrees in \( t \) of all intermediate polynomials.

Example 2.17. In our thread example, after one step in Euclidean algorithm, we obtain:

\[
((2t+1)X^2 + tX) \cdot \tilde{P} + (2t^5 + t^4 + t^3 + 2t^2) \cdot L = (2t^3 + t^2)X^3 + (t^3 + 2t^2 + 1)X^2 + (2t^2 + 2t + 2)X
\]

so that we can take:

\[
U = (2t+1)X^2 + tX, \quad V = 2t^5 + t^4 + t^3 + 2t^2
\]

and \( R = (2t^3 + t^2)X^3 + (t^3 + 2t^2 + 1)X^2 + (2t^2 + 2t + 2)X \).

The next proposition is the key result on which Gao’s algorithm is based.

Proposition 2.18. With the above notations, we have the relation \( R = UP \) where \( P \) is the Ore polynomial we used to construct the codeword \( c \).

Step 3: Left Euclidean division. We compute the quotient \( Q \) in the left Euclidean division of \( R \) by \( U \).

By Proposition 2.18, \( c = \gamma_{k,c,g}(Q) \) and we have decoded the message \( m \).

Example 2.19. In our thread example, the left Euclidean division of \( R \) by \( U \) reads \( R = U \cdot (1 + t^2X) \); we have then reconstructed the Ore polynomial \( P \) we started with.

References


