POLYNOMIAL TIME KEY-RECOVERY ATTACK ON HIGH RATE RANDOM ALTERNANT CODES

MAGALI BARDET, ROCCO MORA, AND JEAN–PIERRE TILLICH

Abstract. A long standing open question is whether the distinguisher of high rate alternant codes or Goppa codes [FGO$^+$11] can be turned into an algorithm recovering the algebraic structure of such codes from the mere knowledge of an arbitrary generator matrix of it. This would allow to break the McEliece scheme as soon as the code rate is large enough and would break all instances of the CFS signature scheme. We give for the first time a positive answer for this problem when the code is *a generic alternant code* and when the code field size q is small : $q \in \{2, 3\}$ and for *all* regime of other parameters for which the aforementioned distinguisher works. This breakthrough has been obtained by two different ingredients :
(i) a way of using code shortening and the component-wise product of codes to derive from the original alternant code a sequence of alternant codes of decreasing degree up to getting an alternant code of degree 3 (with a multiplier and support related to those of the original alternant code);
(ii) an original Gröbner basis approach which takes into account the non standard constraints on the multiplier and support of an alternant code which recovers in polynomial time the relevant algebraic structure of an alternant code of degree 3 from the mere knowledge of a basis for it.

1. Introduction

The McEliece scheme. The McEliece encryption scheme [McE78], which dates back to 1978 and which is almost as old as RSA [RSA78], is a code-based cryptosystem built upon the family of binary Goppa codes. We will denote this system by McEliece-binary Goppa from now on since we will be interested in variations of the McEliece cryptosystem obtained by changing the underlying code family. It is equipped with very fast encryption and decryption algorithms and has very small ciphertexts but large public keysize. Contrarily to RSA which is broken by quantum computers [Sho94], it is also widely viewed as a viable quantum safe cryptosystem. A variation of this public key cryptosystem intended to be IND-CCA secure and an associated key exchange protocol [ABC$^+$20] are now in the process of being standardized by the NIST as quantum safe cryptosystems. It should be noted that the best quantum algorithm for breaking it [KT17] has exponential complexity and the corresponding exponent barely improves the exponent of the best classical algorithm [BM17] by about 40 percent.

The attacks on the McEliece scheme can be put in two classes: message-recovery attacks on one hand and key-recovery attacks on the other hand. The first one consists in inverting the McEliece encryption without finding a trapdoor and makes use of generic decoding algorithms. Despite considerable improvements [Ste88, CC98, MMT11, BJMM12, MO15, BM17], all these algorithms have exponential complexity. The second one consists in trying to recover the private key or an

equivalent private key which allows to decipher as efficiently as a legitimate user. The most efficient attack of this kind was given in [LS01] and essentially consists in trying all Goppa polynomials and all possible supports, where the verification consists in solving a code equivalence problem which is often easy with the help of the support splitting algorithm [Sen00]. The complexity of the second attack is also exponential with an exponent which is much bigger than the one obtained for the first attack. This is why the first kind of attack is considered as the main threat against McEliece-Goppa, and consequently the parameters of all those schemes based on Goppa codes have been chosen to thwart the first attack.

Towards efficient key recovery attacks. This picture began to change due on one hand to variations on the original McEliece proposal based on binary Goppa codes of rate close to $\frac{1}{2}$ by either considering very high rate binary Goppa codes for devising signature schemes [CFS01], or by moving from binary Goppa codes to nonbinary Goppa codes over large alphabets [BLP10, BLP11], thus decreasing significantly the extension degree m over which the (secret) support of the Goppa code is defined. Recall here that the Goppa code is defined over some finite field $\mathbb{F}_q$ whereas the support of the Goppa code is defined over an extension field $\mathbb{F}_{q^m}$ (see Definition 6 which follows). On the other hand, more structured versions of the McEliece system also appeared, based on quasi-cyclic codes such as [BCGO09, CBB$^+$17] or on quasi-dyadic codes such as [MB09, BLM11, BBB$^+$17].

The quasi-cyclic or quasi-dyadic Goppa codes could be attacked by an algebraic modeling [FOPT10, GUL09] for the secret key which could be efficiently solved with Gröbner bases techniques because the added structure allowed to reduce drastically the number of unknowns of the algebraic system. By trying to solve the same algebraic system in the case of high rate Goppa codes it was also found that Gröbner bases techniques behaved very differently when the system corresponds to a Goppa code instead of a random linear code of the same length and dimension. This approach lead to [FGO$^+$11] that gave a way to distinguish high rate Goppa codes from random codes. It is based on the kernel of a linear system related to the aforementioned algebraic system. Indeed, it was shown to have an unexpectedly high dimension when instantiated with Goppa codes or the more general family of alternant codes rather than with random linear codes. This distinguisher was later on given another interpretation in [MP12], where it was proved that this dimension is related to the dimension of the square of the dual of the Goppa code. Very recently, [MT21] revisited [FGO$^+$11] and gave rigorous bounds for the dimensions of the square codes of Goppa or alternant codes and a better insight on the algebraic structure of these squares. Recall here that the component-wise product of codes (also called the Schur product) is defined from the component-wise product of vectors $\boldsymbol{a} = (a_i)_{1\leqslant i\leqslant n}$ and $\boldsymbol{b} = (b_i)_{1\leqslant i\leqslant n}$

$$\boldsymbol{a} \star \boldsymbol{b} \stackrel{\text{def}}{=} (a_1 b_1, \ldots, a_n b_n)$$

by

Definition 1. The *component-wise product of codes* $\mathscr{C}, \mathscr{D}$ over $\mathbb{F}$ with the same length n is defined as

$$\mathscr{C} \star \mathscr{D} \stackrel{\text{def}}{=} \langle\, \boldsymbol{c} \star \boldsymbol{d} \mid \boldsymbol{c} \in \mathscr{C}, \boldsymbol{d} \in \mathscr{D} \,\rangle_{\mathbb{F}}\,.$$

If $\mathscr{C} = \mathscr{D}$, we call $\mathscr{C}^{\star 2} \stackrel{\text{def}}{=} \mathscr{C} \star \mathscr{C}$ the *square code* of $\mathscr{C}$.

Square code and cryptanalysis. Interestingly enough, it was proved in [CGG$^+$14] that square code considerations could also be used to mount an attack on McEliece or Niederreiter schemes based on Generalized Reed-Solomon (GRS) codes. Recall that this scheme was proposed in [Nie86] and was subsequently broken in [SS92]. Note that when the extension degree of the Goppa code is 1 (i.e. the support of the Goppa code is defined over the same field as the Goppa code itself), a Goppa code is indeed a GRS code, so a McEliece scheme based on a Goppa code of extension degree 1 can be attacked with the [SS92] attack. However, this does not seem to generalize to higher extension degrees; i.e. on McEliece schemes based on Goppa codes in general. The point of the new attack [CGG$^+$14], is that it uses arguments on square codes for which there is hope that they could be applied to a much broader class of Goppa codes. It is insightful here to recall in a simplified way the attack obtained in [CGG$^+$14, §5].

Indeed, GRS codes turn out to display a very peculiar property with respect to the square of codes. It is readily seen that $\dim \mathscr{C}^{\star 2} \leqslant \min\left(n, \binom{k+1}{2}\right)$ where k and n are respectively the dimension and length of $\mathscr{C}$. It turns out that for random codes the upper-bound is almost always an equality [CCMZ15], whereas the situation for GRS codes is completely different: in this case, we namely have

$$\dim \mathscr{C}^{\star 2} = \min\left(n, 2k-1\right). \tag{1}$$

The reason of this peculiar behavior comes from the fact that GRS codes are polynomial evaluation codes. Indeed, recall that a GRS code is given by

Definition 2 (GRS code)**.** Let $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{F}^n$ be a vector of pairwise distinct entries and $\boldsymbol{y} = (y_1, \ldots, y_n) \in \mathbb{F}^n$ a vector of nonzero entries. The $[n, k]$ *generalized Reed-Solomon (GRS) code* with *support* $\boldsymbol{x}$ and *multiplier* $\boldsymbol{y}$ is

$$\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y}) \stackrel{\text{def}}{=} \{(y_1 P(x_1), \ldots, y_n P(x_n)) \mid P \in \mathbb{F}[z], \deg P < k\}.$$

Since the Schur product of two polynomial evaluations of degree $\deg P \leqslant k-1$ and $\deg Q \leqslant k-1$ respectively is itself a polynomial evaluation of degree $\deg(P \cdot Q) = \deg P + \deg Q < 2k-1$: $(y_i P(x_i))_i \star (y_i Q(x_i))_i = (y_i^2 P \cdot Q(x_i))_i$, it is readily seen that

$$\mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})^{\star 2} = \mathbf{GRS}_{2k-1}(\boldsymbol{x}, \boldsymbol{y} \star \boldsymbol{y}), \tag{2}$$

which explains (1). In a sense, the square code construction "sees" the polynomial structure of the GRS code. A key recovery attack could be mounted as follows. Recall that it amounts here to recover from an arbitrary generator matrix of a GRS code $\mathscr{C} = \mathbf{GRS}_k(\boldsymbol{x}, \boldsymbol{y})$ a pair $(\boldsymbol{x}', \boldsymbol{y}')$ satisfying $\mathscr{C} = \mathbf{GRS}_k(\boldsymbol{x}', \boldsymbol{y}')$. Let us define $\mathscr{C}(i)$ as the subcode of the GRS code $\mathscr{C}$ given by

$$\mathscr{C}(i) = \{(y_i P(x_j))_{1 \leqslant j \leqslant n} : \deg P < k,\ x_1 \text{ is a zero of order} \geqslant i \text{ of } P\},$$

then

(i) $\mathscr{C}(1)$ can be readily computed from $\mathscr{C}$ since it is the shortened code of $\mathscr{C}$ in the first position.

(ii) We have in general the equality

$$\mathscr{C}(i-1) \star \mathscr{C}(i+1) = \mathscr{C}(i)^{\star 2} \tag{3}$$

coming from the fact that the product of two polynomials which have a zero of order i at x_1 gives a polynomial with a zero of order $2i$ in x_1 and

so does the product of a polynomial with a zero of order $i-1$ in x_1 with a polynomial which has a zero of order $i+1$ at the same place.

(iii) Solving the equation $\mathscr{X} \star \mathscr{A} = \mathscr{B}$ for two known linear codes $\mathscr{A}$ and $\mathscr{B}$ amounts to solve a linear system in the case where $\mathscr{X}$ is the maximal code satisfying $\mathscr{X} \star \mathscr{A} \subseteq \mathscr{B}$. This is indeed the case here for $\mathscr{A} = \mathscr{C}(i-1)$ and $\mathscr{B} = \mathscr{C}(i)^{\star 2}$. $\mathscr{X}$ corresponds in such a case to the *conductor* of $\mathscr{A}$ into $\mathscr{B}$, which can be computed by solving a linear system (see below).

Definition 3. Let $\mathscr{C}, \mathscr{D} \subseteq \mathbb{F}^n$ be two codes. The *conductor* of $\mathscr{C}$ into $\mathscr{D}$ is

$$\mathbf{Cond}\,(\mathscr{C}, \mathscr{D}) \stackrel{\text{def}}{=} \{\boldsymbol{u} \in \mathbb{F}_q^n \mid \boldsymbol{u} \star \mathscr{C} \subseteq \mathscr{D}\},$$

where $\boldsymbol{u} \star \mathscr{C} \stackrel{\text{def}}{=} \{\boldsymbol{u} \star \boldsymbol{c} \mid \boldsymbol{c} \in \mathscr{C}\}$.

Proposition 4 ([CMCP17, Lemma 7])**.** *The conductor can be computed with*

$$\mathbf{Cond}\,(\mathscr{C}, \mathscr{D}) = (\mathscr{C} \star \mathscr{D}^{\perp})^{\perp}.$$

The first two points show that we can therefore compute $\mathscr{C}(2)$ in polynomial time, because $\mathscr{C}(1)$ and $\mathscr{C}(0)$ are known (the first is the shortened code and the second is the code $\mathscr{C}$ itself). We can iterate this process and compute recursively the decreasing set of codes $\mathscr{C}(3), \mathscr{C}(4), \cdots$ and stop when we get a code of dimension 1 (i.e. $C(k-1)$) which basically reveals a great deal of information about the multiplier $\boldsymbol{y}$. It is then straightforward with this approach to finish the attack to recover the whole algebraic structure of $\mathscr{C}$. Note that we have basically computed a decreasing set of codes

$$\mathscr{C} = \mathscr{C}(0) \supset \mathscr{C}(1) \supset \mathscr{C}(2) \cdots \supset \mathscr{C}(k-1)$$

that we will call a *filtration* in what follows.

This approach works basically like this to attack a McEliece scheme based on GRS codes [CGG$^+$14], but interestingly enough it also applies to Wild Goppa codes of extension degree 2 as shown in [COT14]. Such schemes were indeed proposed in [BLP10]. This extension degree corresponds to the largest extension degree where we can expect the Goppa code to behave differently from a random linear code with respect to the square code dimension. Roughly speaking in this case, even if Goppa codes are subfield subcodes of GRS codes, the equality (3) "almost" holds and this is sufficient to mount a similar attack. However as explained in [COT14], this approach is bound to fail when the extension degree m is bigger than 2. However as observed in [MP12], even when $m > 2$, the square code $\mathscr{C}^{\star 2}$ can also be of unusually small dimension when the rate of the Goppa code is close to 1, but this time not by taking $\mathscr{C}$ to be the Goppa code itself, but by choosing $\mathscr{C}$ to be the *dual* of the Goppa code. This strongly suggests that an approach similar to [CGG$^+$14, COT14] could be followed to attack McEliece schemes based on very high rate Goppa codes. Even if the parameters of the McEliece schemes proposed in the literature are never in the regime where the dimension of the square of the dual of the Goppa code behaves differently from a random linear code, there is the notable exception of the code-based signature scheme [CFS01], which is based in a crucial way on high rate Goppa codes, and which similarly to the McEliece scheme would be broken, if we can recover the unknown support of the Goppa code from an arbitrary generator matrix for it. However, the fact that the dual code is actually the trace code of a GRS code but not a subfield subcode of a GRS code loses a lot of the original polynomial structure and seems to complicate very significantly this

approach. This is still an open problem since the problem was explicitly raised in [FGO$^+$11].

Our contribution. In the present article, we make what we consider to be a significant step in this direction. We will namely show that somewhat unexpectedly, an equality related to (3) holds, when taking duals of *(generic) high rate alternant codes*, but not when we take Goppa codes. This is extremely surprising because Goppa codes are just alternant codes with a peculiar structure. To explain the phenomenon we are witnessing, let us recall the definition of an alternant code which is a subfield subcode of a GRS code:

Definition 5 (alternant code)**.** Let $n \leqslant q^m$, for some positive integer m. Let $\mathbf{GRS}_r(\boldsymbol{x}, \boldsymbol{y})$ be the GRS code over $\mathbb{F}_{q^m}$ of dimension r with support $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ and multiplier $\boldsymbol{y} \in (\mathbb{F}_{q^m} \backslash \{0\})^n$. The *alternant code* with support $\boldsymbol{x}$ and multiplier $\boldsymbol{y}$ and *degree* r over $\mathbb{F}_q$ is

$$\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y}) \stackrel{\text{def}}{=} \mathbf{GRS}_r(\boldsymbol{x}, \boldsymbol{y})^\perp \cap \mathbb{F}_q^n = \mathbf{GRS}_{n-r}(\boldsymbol{x}, \boldsymbol{y}^\perp) \cap \mathbb{F}_q^n,$$

where

$$\boldsymbol{y}^\perp \stackrel{\text{def}}{=} \left(\frac{1}{\pi'_{\boldsymbol{x}}(x_1) y_1}, \ldots, \frac{1}{\pi'_{\boldsymbol{x}}(x_n) y_n} \right)$$

and $\pi'_{\boldsymbol{x}}$ is the derivative of $\pi_{\boldsymbol{x}}$. The integer m is called *extension degree* of the alternant code.

In other words, an alternant code is the dual (over $\mathbb{F}_q$) of a GRS code defined over $\mathbb{F}_{q^m}$. It is also a subfield subcode of a GRS code since the dual of a GRS code is a GRS code : $\mathbf{GRS}_r(\boldsymbol{x}, \boldsymbol{y})^\perp = \mathbf{GRS}_{n-r}(\boldsymbol{x}, \boldsymbol{y}^\perp)$, see [MS86, Theorem 4, p. 304]. We also recall that Goppa codes form a particular family of alternant codes

Definition 6 (Goppa code)**.** Let $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ be a support vector and $\Gamma \in \mathbb{F}_{q^m}[z]$ a polynomial of degree r such that $\Gamma(x_i) \neq 0$ for all $i \in \{1, \ldots, n\}$. The *Goppa code* of degree r with support $\boldsymbol{x}$ and *Goppa polynomial* Γ is defined as

$$\mathscr{G}(\boldsymbol{x}, \Gamma) \stackrel{\text{def}}{=} \mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y}),$$

where $\boldsymbol{y} \stackrel{\text{def}}{=} \left(\frac{1}{\Gamma(x_1)}, \ldots, \frac{1}{\Gamma(x_n)} \right)$.

The very unusual behavior we observe in the case of a generic alternant code $\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$ is that in a certain high rate regime, if we shorten its *dual* in one position i and take its square to get $\mathscr{B} \stackrel{\text{def}}{=} \left(\mathbf{Sh}_i \left(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})^\perp \right) \right)^{\star 2}$, where $\mathbf{Sh}_i(\mathscr{C})$ denotes the code $\mathscr{C}$ shortened in position i, then the maximal code $\mathscr{X}$ satisfying

$$\mathscr{X} \star \mathscr{A} \subseteq \mathscr{B}$$

is the dual of a certain alternant code of degree $r-1$:

$$\mathscr{X} = \mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}}, \boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}} - x_i))^\perp.$$

where $\boldsymbol{x}_{\check{i}}$ denotes the vector $\boldsymbol{x}$ where we have dropped the index i and $\mathscr{A}$ is the dual of the shortened alternant code in position i, i.e. $(\mathbf{Sh}_i(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})))^\perp$. We also use the short notation $\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}} - x_i)$ for $\boldsymbol{y}_{\check{i}} \star (x_j - x_i)_{j \neq i}$. Note that this code is actually the dual of an alternant code since $(\mathbf{Sh}_i(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})))^\perp = \mathscr{A}_r(\boldsymbol{x}_{\check{i}}, \boldsymbol{y}_{\check{i}})^\perp$ (see Proposition 12). In other words

$$\mathbf{Cond}\left(\mathscr{A}_r(\boldsymbol{x}_{\check{i}}, \boldsymbol{y}_{\check{i}}), \left(\mathbf{Sh}_i \left(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})^\perp \right) \right)^{\star 2} \right) = \mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}}, \boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}} - x_i))^\perp. \tag{4}$$

This means that starting from a generic alternant code $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ of degree r, we can derive in polynomial time, by first computing two auxiliary codes $\mathscr{A} = (\mathbf{Sh}_i(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})))^{\perp}$ and $\mathscr{B} = \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)^{\star 2}$, and then computing the conductor $\mathbf{Cond}(\mathscr{A},\mathscr{B})$ of $\mathscr{A}$ into $\mathscr{B}$, an alternant code of degree $r-1$. As we will see, there are only two conditions to be met for performing this task:

(i) $r \geqslant q+1$ where q is the alphabet size of the alternant code,
(ii) $\left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)^{\star 2}$ is not the full code $\mathbb{F}_q^{n-1}$ where n is the codelength of the alternant code.

By iterating this process, we can compute in polynomial time some kind of "filtration" of duals of alternant codes of decreasing degree

$$\mathscr{A}_r^{\perp} = \mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp} \overset{\mathbf{Sh}_{i_1}}{\supseteq} \mathscr{A}_{r-1}^{\perp} \overset{\mathbf{Sh}_{i_2}}{\supseteq} \cdots \overset{\mathbf{Sh}_{i_{r-q}}}{\supseteq} \mathscr{A}_q{}^{\perp} \tag{5}$$

with multipliers and support which are related to the original support and multiplier (and from which the original support and multiplier can be easily recovered). Here the notation $\mathscr{A} \overset{\mathbf{Sh}_i}{\supseteq} \mathscr{B}$ means that

$$\mathbf{Sh}_i(\mathscr{A}) \supseteq \mathscr{B}.$$

What can we do with this sequence? The point is that if the degree of the alternant code is small enough, we can compute its support and multiplier by solving a low degree algebraic system related to the algebraic systems considered in [FOPT10, FGO$^+$13]. We will detail this in the particular case where $r = 3$ and show that in this case, solving the system can be performed in polynomial time with Gröbner basis techniques. Roughly speaking the reason for this is that we have a conjunction of factors : a very overdetermined and highly structured system which gives during the Gröbner basis computation many new very low degree equations. We will also show that it is possible to speed up significantly the system solving process by introducing in the algebraic modeling new low degree polynomial equations which are not in the ideal of the original algebraic equations from [FGO$^+$13] and which express the fact that the multiplier vector has only non zero entries and the support vector has only distinct entries. This will result in the end in a very efficient system solving procedure. Note that the aforementioned procedure reaches an alternant code $\mathscr{A}_3$ of degree 3 when the field size q is either equal to 2 or 3. In other words, we have at the end a way to break a McEliece scheme based on *binary or ternary alternant codes* as soon as $\left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)^{\star 2}$ is not the full code $\mathbb{F}_q^{n-1}$. By using the formula given in [MT21], this is the case when

$$\begin{aligned} n-1 \quad &> \quad \binom{rm+1}{2} - \frac{m}{2}(r-1)\left((2e+1)r - 2\frac{q^e-1}{q-1}\right), \\ &\text{where } e \quad \overset{\text{def}}{=} \quad \max\{i \in \mathbb{N} \mid r \geqslant q^i+1\} = \left\lfloor \log_q(r-1) \right\rfloor \end{aligned} \tag{6}$$

We give in Table 1 the known cases where it is possible to attack a McEliece scheme based on alternant codes together with the new attack proposed here:

In a nutshell, our contribution can be summarized as follows.

Breaking the $m = 2$ barrier. It is has been a long standing open problem after the [SS92] attack on McEliece-GRS whether it is also possible to attack subfield subcodes of GRS codes, i.e. attack McEliece-alternant or McEliece-Goppa. A first step in this direction was made in [CGG$^+$14] where a new attack on McEliece-GRS was derived with a hope to generalize it to McEliece-alternant or McEliece-Goppa

paper	restriction
[SS92, CGG$^+$14]	$m = 1$
[COT14]	$m = 2$ + Wild Goppa code
this paper	$q = 2$ or $q = 3$, m arbitrary + high rate condition (6) (does not apply in the particular case of Goppa codes)

TABLE 1. Summary of polynomial time attacks on McEliece schemes based on alternant codes with the conditions to apply them.

because it is in essence only based on the fact that certain alternant or Goppa codes behave differently from random codes with respect to the dimension of the square code. This was confirmed in [COT14] by attacking McEliece-wild Goppa in the particular case where the extension degree m is 2, but the method used there which uses squares of the (shortened) Goppa code is bound to fail for higher extension degrees. Here we break for the first time the $m = 2$ barrier, which was even conjectured at some point to be the ultimate limit for such algebraic attacks to work in polynomial time and show that we can actually attack McEliece-alternant for *any* extension degree m provided that the rate of the alternant code is sufficiently large (6) and the field size sufficiently low $q = 2$ or $q = 3$. Our attack is also based on square code considerations, but this time on the *dual* of the alternant code. The point is that in this case the square of the dual can also be distinguished from a random code in this regime [FGO$^+$11, MP12]. The attack is however much more involved in this case, because the dual loses the simple polynomial evaluation formulation of the Goppa code, since it is in this case the trace of a GRS code and not subfield subcode of a GRS code. Understanding the structure of the square is more complicated as was already apparent in [MT21] which started such a task.

Opening the road for attacking the CFS scheme. Interestingly our attack does not work at all when the alternant code has the additional structure of being a Goppa code. However this work could open the road for also attacking this subcase, in which case we could hope to break the CFS scheme [CFS01] which operates precisely in the high rate regime where the square of the dual of the Goppa code behaves abnormally.

New algebraic attack. Our attack consists in two phases, the first phase computes a filtration of the dual of the alternant code by computing iteratively conductors and the second phase solves with Gröbner bases techniques a variation of the algebraic system considered in [FGO$^+$13] and recovers the support and the multiplier from the dual of the alternant code of degree 3 we have at the end of the filtration when $q = 2$ or $q = 3$. We improve rather significantly upon the complexity of solving this system by adding new equations expressing the constraints on the support (all elements are distinct) and the multipliers (all elements are nonzero). By using certain heuristics that we confirmed experimentally we are able to prove that the Gröbner basis computation takes polynomial time and give a complete algebraic explanation of each step of the computation. It is likely that this analysis could be carried over for larger constant degree alternant codes. This would allow to break McEliece-alternant for larger field size than 3.

A proof-of-concept implementation in MAGMA of the whole attack can be found at `https://github.com/roccomora/HighRateAlternant`.

2. Notation and prerequisites

This section fixes the notation and reviews some tools and constructions needed for understanding the filtration attack and the rest of the article. These include the notion of subfield subcodes, trace codes, puncturing and shortening operators.

2.1. Notation.

Finite fields. We denote by $\mathbb{F}$ a generic finite field and we will use it whenever it is not important to specify the field size. We will extensively consider the finite field extension $\mathbb{F}_q \subseteq \mathbb{F}_{q^m}$, where $\mathbb{F}_q$ and $\mathbb{F}_{q^m}$ are two finite fields with q and q^m elements respectively, for a prime power q and a positive integer m.

Vector and matrix notation. Vectors are indicated by lowercase bold letters $\boldsymbol{x}$ and matrices by uppercase bold letters $\boldsymbol{M}$. By convention, vectors coordinates are indexed starting from 1. Moreover, $[\![a, b]\!]$ indicates the closed integer interval between a and b. Given a function f acting on $\mathbb{F}$ and a vector $\boldsymbol{x} = (x_i)_{1 \leqslant i \leqslant n} \in \mathbb{F}$, the expression $f(\boldsymbol{x})$ is the component-wise mapping of f on $\boldsymbol{x}$, i.e. $f(\boldsymbol{x}) = (f(x_i))_{1 \leqslant i \leqslant n}$. We will even apply this with functions f acting on $\mathbb{F} \times \mathbb{F}$: for instance for two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ in $\mathbb{F}^n$ and two positive integers a and b we denote by $\boldsymbol{x}^a\boldsymbol{y}^b$ the vector $(x_i^a y_i^b)_{1 \leqslant i \leqslant n}$. It will also be convenient to use a notation to say that we drop a few positions in a vector. If $\boldsymbol{x} = (x_i)_{i \in [\![1,n]\!]}$ and $\mathcal{I}$ is a subset of positions, we denote by $\boldsymbol{x}_{\overset{\sim}{\mathcal{I}}}$ the vector $\boldsymbol{x}_{\overset{\sim}{\mathcal{I}}} \overset{\text{def}}{=} (x_i)_{i \in [\![1,n]\!] \setminus \mathcal{I}}$. When there is just one position i in $\mathcal{I}$ we simply write $\boldsymbol{x}_{\tilde{i}}$ in this case (for instance, if $\boldsymbol{x} = (x_1, x_2, x_3)$ then $\boldsymbol{x}_{\overset{\sim}{2}} = (x_1, x_3)$).

Vector spaces. Vector spaces are indicated by $\mathscr{C}$. For two vector spaces $\mathscr{C}$ and $\mathscr{D}$, the notation $\mathscr{C} \oplus \mathscr{D}$ means that the two vector spaces are in direct sum, i.e. that $\mathscr{C} \cap \mathscr{D} = \{0\}$.

2.2. Trace codes and subfield subcodes.

A useful $\mathbb{F}_q$-linear map from $\mathbb{F}_{q^m}$ to its subfield $\mathbb{F}_q$ is the *trace operator*:

Definition 7. Given the finite field extension $\mathbb{F}_{q^m}/\mathbb{F}_q$, we define the *field trace* $\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}$ for all $x \in \mathbb{F}_{q^m}$ as the $\mathbb{F}_q$-linear map from $\mathbb{F}_{q^m}$ to $\mathbb{F}_q$ such that

$$\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(x) = \sum_{i=0}^{m-1} x^{q^i}.$$

The definition extends to vectors $\boldsymbol{x} \in \mathbb{F}_{q^m}^n$ so that the trace acts component-wise:

$$\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\boldsymbol{x}) = (\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(x_1), \ldots, \mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(x_n))$$

and hence to codes $\mathscr{C}$ over $\mathbb{F}_{q^m}$

$$\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\mathscr{C}) = \{\mathrm{Tr}_{\mathbb{F}_{q^m}/\mathbb{F}_q}(\boldsymbol{c}) \mid \boldsymbol{c} \in \mathscr{C}\}.$$

We will omit the subscript $\mathbb{F}_{q^m}/\mathbb{F}_q$ and write more coincisely Tr from now on, whenever the extension field is clear from the context. Another standard construction to move from codes over $\mathbb{F}_{q^m}$ to codes over a subfield $\mathbb{F}_q$ is the one of subfield subcode.

Definition 8. Let $\mathscr{C} \subseteq \mathbb{F}_{q^m}^n$ be a code. The code

$$\mathscr{C}_{|\mathbb{F}_q} \overset{\text{def}}{=} \mathscr{C} \cap \mathbb{F}_q^n = \{\boldsymbol{c} \in \mathscr{C} \mid \forall i \in [\![1, n]\!],\, c_i \in \mathbb{F}_q\}$$

is called *subfield subcode* over $\mathbb{F}_q$ of $\mathscr{C}$.

It is easy to see that $\mathscr{C}_{|\mathbb{F}_q}$ is an $\mathbb{F}_q$-linear subspace of $\mathbb{F}_{q^m}^n$, where n is the length of $\mathscr{C}$. We conclude this subsection by stating the main classical result linking trace codes to subfield subcodes, i.e. Delsarte's theorem.

Theorem 9. *(Delsarte's Theorem* [Del75]*) Let $\mathscr{C}$ be a code over $\mathbb{F}_{q^m}$. Then*

$$(\mathscr{C}_{|\mathbb{F}_q})^\perp = Tr_{\mathbb{F}_{q^m}/\mathbb{F}_q}\left(\mathscr{C}^\perp\right).$$

2.3. **Shortening and Puncturing.**

Definition 10. Given a code $\mathscr{C} \subseteq \mathbb{F}^n$ and a subset $\mathcal{I} \subseteq [\![1, n]\!]$, the *punctured* code $\mathbf{Pct}_{\mathcal{I}}(\mathscr{C})$ and the *shortened* code $\mathbf{Sh}_{\mathcal{I}}(\mathscr{C})$ over $\mathcal{I}$ are defined respectively as

$$\mathbf{Pct}_{\mathcal{I}}(\mathscr{C}) = \left\{(c_i)_{i\in[\![1,n]\!]\setminus\mathcal{I}} \mid \boldsymbol{c} \in \mathscr{C}\right\},$$
$$\mathbf{Sh}_{\mathcal{I}}(\mathscr{C}) = \left\{(c_i)_{i\in[\![1,n]\!]\setminus\mathcal{I}} \mid \exists\, \boldsymbol{c} = (c_i)_{i\in[\![1,n]\!]} \in \mathscr{C} \text{ s.t. } \forall\, i \in \mathcal{I}, c_i = 0\right\}.$$

For the sake of simplicity, when $\mathcal{I} = \{i\}$, we denote the punctured and the shortened codes in $\mathcal{I}$ with $\mathbf{Pct}_i(\mathscr{C})$ and $\mathbf{Sh}_i(\mathscr{C})$ respectively.

Since we will frequently move from a code to its dual, it is worthwhile to mention how the shortening and puncturing combine with the dual operator:

Proposition 11. [HP03, Theorem 1.5.7] *Let $\mathscr{C}$ be a linear code of length n and $\mathcal{I} \subset [\![1, n]\!]$. Then*

$$\mathbf{Sh}_{\mathcal{I}}\left(\mathscr{C}^\perp\right) = \mathbf{Pct}_{\mathcal{I}}(\mathscr{C})^\perp \quad \textit{and} \quad \mathbf{Pct}_{\mathcal{I}}\left(\mathscr{C}^\perp\right) = \mathbf{Sh}_{\mathcal{I}}(\mathscr{C})^\perp.$$

Finally we recall the well known fact that a shortened alternant codes is itself an alternant code

Proposition 12. *Let $\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$ be an alternant code of length n and $\mathcal{I} \subseteq \{1, \ldots, n\}$. Then*

$$\mathbf{Sh}_{\mathcal{I}}\left(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})\right) = \mathscr{A}_r(\boldsymbol{x}_{\breve{\mathcal{I}}}, \boldsymbol{y}_{\breve{\mathcal{I}}}).$$

2.4. **Base field extension and alternant codes.** It will be convenient to consider for a code defined over $\mathbb{F}_q$ its extension by scalars over $\mathbb{F}_{q^m}$, meaning the following.

Definition 13 (extension of a code over a field extension)**.** Let $\mathscr{C}$ be a linear code over $\mathbb{F}_q$. We denote by $\mathscr{C}_{\mathbb{F}_{q^m}}$ the $\mathbb{F}_{q^m}$-linear span of $\mathscr{C}$ in $\mathbb{F}_{q^m}^n$.

It could seem that this is a somewhat trivial object, but this notion comes very handy in our case, since the extension to $\mathbb{F}_{q^m}$ of the dual of an alternant code defined over $\mathbb{F}_q$ is a sum of m GRS codes as shown by

Proposition 14. *Let $\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$ be an alternant code over $\mathbb{F}_q$. Then*

$$\left(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})^\perp\right)_{\mathbb{F}_{q^m}} = \sum_{j=0}^{m-1} \mathbf{GRS}_r(\boldsymbol{x}, \boldsymbol{y})^{q^j}.$$

where $\mathscr{C}^{q^j} \overset{def}{=} \{\boldsymbol{c}^{q^j} = (c_i^{q^j})_i : \boldsymbol{c} \in \mathscr{C}\}$ is readily seen to be an $\mathbb{F}_{q^m}$-linear code of the same dimension as $\mathscr{C}$ when $\mathscr{C}$ is itself an $\mathbb{F}_{q^m}$-linear code

Proof. It follows directly from Proposition 9 by taking $\mathscr{C} \overset{\text{def}}{=} \mathbf{GRS}_r(\boldsymbol{x}, \boldsymbol{y})$ and the fact that we have for any nonnegative integer j and any $\mathbb{F}_{q^m}$-linear code $\mathscr{C}$ (see Proposition 41 in Appendix A).

$$\mathrm{Tr}\left(\mathscr{C}\right)_{\mathbb{F}_{q^m}} = \mathscr{C} + \mathscr{C}^q + \cdots + \mathscr{C}^{q^{m-1}}.$$

□

We will also need the following straightforward properties

Lemma 15. [Ran15, Lemma 2.23] *Let $\mathscr{C}, \mathscr{D} \subseteq \mathbb{F}_q^n$ be two codes. Then*

(1) $(\mathscr{C}^\perp)_{\mathbb{F}_{q^m}} = (\mathscr{C}_{\mathbb{F}_{q^m}})^\perp \subseteq \mathbb{F}_{q^m}^n$.
(2) $\mathscr{C} \subseteq \mathscr{D} \iff \mathscr{C}_{\mathbb{F}_{q^m}} \subseteq \mathscr{D}_{\mathbb{F}_{q^m}}$.
(3) $(\mathscr{C} + \mathscr{D})_{\mathbb{F}_{q^m}} = \mathscr{C}_{\mathbb{F}_{q^m}} + \mathscr{D}_{\mathbb{F}_{q^m}}$ *and* $(\mathscr{C} \oplus \mathscr{D})_{\mathbb{F}_{q^m}} = \mathscr{C}_{\mathbb{F}_{q^m}} \oplus \mathscr{D}_{\mathbb{F}_{q^m}}$.
(4) $(\mathscr{C} \cap \mathscr{D})_{\mathbb{F}_{q^m}} = \mathscr{C}_{\mathbb{F}_{q^m}} \cap \mathscr{D}_{\mathbb{F}_{q^m}}$.
(5) $(\mathscr{C} \star \mathscr{D})_{\mathbb{F}_{q^m}} = \mathscr{C}_{\mathbb{F}_{q^m}} \star \mathscr{D}_{\mathbb{F}_{q^m}}$.

2.5. **A few useful notions for algebraic system solving.** We will use here techniques inspired by Gröbner bases computations and it will be useful to recall a few useful notions about this topic. We will consider here ideals generated by polynomials over some field $\mathbb{F}$ in a certain number of variables, say $X_1, \ldots, X_n$. We will use two orders on the monomials of $\mathbb{F}[X_1, \cdots, X_n]$. The first is just the usual lexicographic order where the single variables are ordered as $X_1 > X_2 > \cdots > X_n$ and two monomials $X^\alpha \stackrel{\text{def}}{=} X_1^{\alpha_1} \cdots X_n^{\alpha_n}$ and $X^\beta \stackrel{\text{def}}{=} X_1^{\beta_1} \cdots X_n^{\alpha_n}$ are ordered as follows: $X^\alpha > X^\beta$ if and only if there exists $i \in [\![1, n]\!]$ such that $\alpha_i > \beta_i$ and $\alpha_j = \beta_j$ for $j < i$. We will also use the *graded reverse lexicographical order (grevlex)* meaning that $X^\alpha > X^\beta$ if and only if there exists $i \in [\![1, n]\!]$ such that $\alpha_i < \beta_i$ and $\alpha_j = \beta_j$ for $j > i$. We will also need the notion of *leading monomial*. The leading monomial $\mathrm{LM}(P)$ of a polynomial (for a given order on the monomials) is the largest monomial appearing in P. In other words for $P = \sum_{\alpha \in \mathbb{Z}^n} c_\alpha X^\alpha$ with $P \in \mathbb{F}[X_1, \cdots, X_n]$ where we use the notation $X^\alpha = X_1^{\alpha_1} \cdots X_n^{\alpha_n}$ we have $\mathrm{LM}(P) = \max\{X^\alpha \mid c_\alpha \neq 0\}$ where the maximum is taken with respect to the order on the monomials which is chosen.

Another notion will be useful for manipulating an algebraic system, namely

Definition 16. The Macaulay $\mathcal{M}^{\text{acaulay}}(f_1, \cdots, f_k)$ matrix associated to an algebraic system (and with respect to a certain order of the monomials) specified by the polynomials $(f_1, \ldots, f_k)$ is the $k \times M$ matrix where M is the number of different monomials $m_1 > m_2 \cdots > m_M$ involved in the f_i's where the entry m_{ij} is the coefficient of the monomial m_j in f_i.

3. The filtration

Assume that $r \geqslant q + 1$. We are going to explain in this section how, starting with the code $\mathscr{A}_r = \mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$, we are (generally) able to compute in polynomial time a sequence of alternant codes such that

$$\mathscr{A}_r^\perp = \mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})^\perp \overset{\mathbf{Sh}_{i_1}}{\supseteq} \mathscr{A}_{r-1}^\perp \overset{\mathbf{Sh}_{i_2}}{\supseteq} \cdots \overset{\mathbf{Sh}_{i_{r-q}}}{\supseteq} \mathscr{A}_q{}^\perp, \tag{7}$$

where all the alternant codes have a support which is easily derived from the support of $\mathscr{A}_r$, since it just amounts to drop some positions. This is instrumental for recovering efficiently the algebraic structure of the alternant code (i.e. the support $\boldsymbol{x}$ and the multiplier $\boldsymbol{y}$) in what follows. The core of this attack is the following theorem.

Theorem 17. *Let $\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})$ be an alternant code such that $r \geqslant q + 1$. Let $\mathscr{C} \stackrel{def}{=} (\mathbf{Sh}_i(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})))^\perp$, $\mathscr{D} \stackrel{def}{=} \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})^\perp\right)\right)^{\star 2}$, for an arbitrary position i. Then*

$$\mathscr{A}_{r-1}(\boldsymbol{x}_{\tilde{i}}, \boldsymbol{y}_{\tilde{i}}(\boldsymbol{x}_{\tilde{i}} - x_i))^\perp \star \mathscr{C} \subseteq \mathscr{D},$$

or, equivalently,

$$\mathbf{Cond}(\mathscr{C}, \mathscr{D}) \supseteq \mathscr{A}_{r-1}(\boldsymbol{x}_{\tilde{i}}, \boldsymbol{y}_{\tilde{i}}(\boldsymbol{x}_{\tilde{i}} - x_i))^\perp.$$

Experimentally it turns out that we actually have equality here $\mathbf{Cond}\left(\mathscr{C},\mathscr{D}\right) = \mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^{\perp}$ when choosing a random alternant code. It is however possible to build artificial examples where equality does not hold. Notably, we also found that the subfamily of Goppa codes does not meet this property either. However, if $\boldsymbol{x}$ and $\boldsymbol{y}$ are sampled at random, we never met a case in our experiments where equality does not hold. This leads us to state the following conjecture.

Conjecture 18. *Let $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ be a random alternant code over $\mathbb{F}_q$, such that $r \geqslant q+1$ and $\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)^{\star 2}$ is not the full code. Let $\mathscr{C} \stackrel{def}{=} \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})\right)\right)^{\perp}$ and $\mathscr{D} \stackrel{def}{=} \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)^{\star 2}$, for an arbitrary position i. Then, we expect that*

$$\mathbf{Cond}\left(\mathscr{C},\mathscr{D}\right) = \mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^{\perp}.$$

with probability $1-o(1)$ as $m,n \to \infty$. **[mb: who tends to ∞? rather "overwhelming probability" than "1-o(1)" ?]**

It is clear that this conjecture, if true, allows to compute in polynomial time the filtration (7), since computing conductors just amounts to solve a linear system. Moreover, if $\boldsymbol{x},\boldsymbol{y}$ are taken at random, then we expect a random behaviour for $\boldsymbol{x}_{\check{i}}$ and $\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i)$ as well. The fact that when taking a random alternant code, the whole filtration can be computed has indeed been verified experimentally. The first conductor $\mathscr{A}_{r-1} = \mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}_1},\boldsymbol{y}_{\check{i}_1}(\boldsymbol{x}_{\check{i}_1}-x_{i_1}))^{\perp}$ is computed by using directly Conjecture 18 and we iterate the process by choosing a sequence of positions i_1, $i_2,\cdots,i_{r_q}$ by which we shorten. We let $\mathcal{I}_s = \{i_1,\cdots,i_s\}$. It is readily seen that we compute iteratively from $\mathscr{A}_{r-s+1}^{\perp} \stackrel{\text{def}}{=} \mathscr{A}_{r-s+1}(\boldsymbol{x}_{\widetilde{\mathcal{I}_{s-1}}},\boldsymbol{y}_{\widetilde{\mathcal{I}_{s-1}}}\prod_{j=1}^{s-1}(\boldsymbol{x}_{\widetilde{\mathcal{I}_{s-1}}}-x_{i_j}))^{\perp}$ the code

$$\mathscr{A}_{r-s}^{\perp} \stackrel{\text{def}}{=} \mathscr{A}_{r-s}(\boldsymbol{x}_{\widetilde{\mathcal{I}_s}},\boldsymbol{y}_{\widetilde{\mathcal{I}_s}}\prod_{j=1}^{s}(\boldsymbol{x}_{\widetilde{\mathcal{I}_s}}-x_{i_j}))^{\perp}.$$

This allows to decrease the degree of the alternant one by one. The last step ends by using the conjecture with $r=q+1$ and ends with the conductor $\mathscr{A}_q^{\perp}$. Let us now prove Theorem 17.

3.1. **Proof of Theorem 17.** It will be convenient to prove a slightly stronger result which implies Theorem 17. It is based on the observation that $\mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^{\perp} \subseteq \mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)$. Theorem 17 is indeed implied by the following slightly stronger result:

Theorem 19. *Let $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ be an alternant code such that $r \geqslant q+1$. Let $\mathscr{C} \stackrel{def}{=} \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})\right)\right)^{\perp}$ and $\mathscr{D}' \stackrel{def}{=} \mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^{\perp} \star \mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)$, for an arbitrary position i. Then*

$$\mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^{\perp} \star \mathscr{C} \subseteq \mathscr{D}',$$

or, equivalently,

$$\mathbf{Cond}\left(\mathscr{C},\mathscr{D}'\right) \supseteq \mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^{\perp}.$$

To prove this theorem, it will be convenient to consider the extensions of all these codes to $\mathbb{F}_{q^m}$, in other words we are going to prove that

$$\mathscr{B}_{\mathbb{F}_{q^m}} \star \mathscr{C}_{\mathbb{F}_{q^m}} \subseteq \mathscr{D}'_{\mathbb{F}_{q^m}}, \tag{8}$$

where $\mathscr{B} \stackrel{\text{def}}{=} \mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^{\perp}$. The point of doing this, is that (i) it is equivalent to prove (8) because of the points 2 and 5 of Lemma 15, (ii) the extended

codes can be expressed as a sum of GRS codes due to Proposition 14. The proof of Theorem 19 will proceed by following the steps below.

Step 1: We first observe that the code $\mathscr{C}_{\mathbb{F}_{q^m}} = \left((\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})\right))^{\perp}\right)_{\mathbb{F}_{q^m}} = \left(\mathscr{A}_r(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}})^{\perp}\right)_{\mathbb{F}_{q^m}}$ (where the last equality follows from Proposition 12) decomposes as

$$\left(\mathscr{A}_r(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}})^{\perp}\right)_{\mathbb{F}_{q^m}} = \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)_{\mathbb{F}_{q^m}} \oplus \left\langle \boldsymbol{y}_{\check{i}} \right\rangle_{\mathbb{F}_{q^m}}.$$

This is Lemma 20 below. This implies that

$$\mathscr{B}_{\mathbb{F}_{q^m}} \star \mathscr{C}_{\mathbb{F}_{q^m}} = \underbrace{\mathscr{B}_{\mathbb{F}_{q^m}} \star \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)_{\mathbb{F}_{q^m}}}_{\mathscr{D}'_{\mathbb{F}_{q^m}}} + \mathscr{B}_{\mathbb{F}_{q^m}} \star \left\langle \boldsymbol{y}_{\check{i}} \right\rangle_{\mathbb{F}_{q^m}}.$$

Therefore in order to prove (8) it will be enough to prove the inclusion

$$\mathscr{B}_{\mathbb{F}_{q^m}} \star \left\langle \boldsymbol{y}_{\check{i}} \right\rangle_{\mathbb{F}_{q^m}} \subseteq \mathscr{D}'_{\mathbb{F}_{q^m}}. \tag{9}$$

Step 2: To achieve this purpose, we then prove that the extended shortened code $\left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)_{\mathbb{F}_{q^m}}$ contains as a subcode $\mathscr{B}_{\mathbb{F}_{q^m}} = \left(\mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},(\boldsymbol{x}_{\check{i}}-x_i)\boldsymbol{y}_{\check{i}})^{\perp}\right)_{\mathbb{F}_{q^m}}$ (Lemma (21)) on one hand and $\mathscr{C}_0 \stackrel{\text{def}}{=} \left\langle y_i^{q^u}\boldsymbol{y}_{\check{i}}^{q^v} - y_i^{q^v}\boldsymbol{y}_{\check{i}}^{q^u} \,:\, u,v \in [\![0,m-1]\!] \right\rangle_{\mathbb{F}_{q^m}}$ on the other hand. Actually more is true, namely that the extended shortened code is a sum of these two subcodes but we will not need this.

Step 3: By using this, in order to prove (9) we will start with an element in $\mathscr{B}_{\mathbb{F}_{q^m}} \star \left\langle \boldsymbol{y}_{\check{i}} \right\rangle_{\mathbb{F}_{q^m}}$ and by adding suitable elements of $\mathscr{B}_{\mathbb{F}_{q^m}} \star \mathscr{C}_0$ and $\mathscr{B}^{\star 2}_{\mathbb{F}_{q^m}}$ we will show that we end with an element in $\mathscr{D}'_{\mathbb{F}_{q^m}} = \mathscr{B}_{\mathbb{F}_{q^m}} \star \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)_{\mathbb{F}_{q^m}}$.

Let us now state and prove the lemmas we have mentioned above.

Lemma 20.

$$\left(\mathscr{A}_r(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}})^{\perp}\right)_{\mathbb{F}_{q^m}} = \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)_{\mathbb{F}_{q^m}} \oplus \left\langle \boldsymbol{y}_{\check{i}} \right\rangle_{\mathbb{F}_{q^m}}.$$

Proof. Note that $\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)_{\mathbb{F}_{q^m}}$ decomposes as the set of codewords $\mathscr{A}_0$ that are zero in i (this is $\left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)_{\mathbb{F}_{q^m}}$ where add an extra-position at i which is always 0) plus a space of dimension 1 generated by an element of $\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)_{\mathbb{F}_{q^m}}$ which is not equal to 0 at position i. $\boldsymbol{y}$ is clearly such an element and we can write

$$\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)_{\mathbb{F}_{q^m}} = \mathscr{A}_0 \oplus \left\langle \boldsymbol{y} \right\rangle_{\mathbb{F}_{q^m}}.$$

By puncturing these codes at i we get our lemma. □

Lemma 21. *We have for any position i*

$$\left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)_{\mathbb{F}_{q^m}} \supseteq \left(\mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},(\boldsymbol{x}_{\check{i}}-x_i)\boldsymbol{y}_{\check{i}})^{\perp}\right)_{\mathbb{F}_{q^m}} \tag{10}$$

$$\left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)\right)_{\mathbb{F}_{q^m}} \supseteq \mathscr{C}_0 \quad \textit{where} \tag{11}$$

$$\mathscr{C}_0 \stackrel{def}{=} \left\langle y_i^{q^u}\boldsymbol{y}_{\check{i}}^{q^v} - y_i^{q^v}\boldsymbol{y}_{\check{i}}^{q^u} \;\middle|\; u,v \in [\![0,m-1]\!] \right\rangle_{\mathbb{F}_{q^m}}. \tag{12}$$

Proof. By using Proposition 14 we know that

$$\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^{\perp}\right)_{\mathbb{F}_{q^m}} = \left\langle \boldsymbol{x}^{aq^\ell}\boldsymbol{y}^{q^\ell};\ a \in [\![0,r-1]\!],\ \ell \in [\![0,m-1]\!] \right\rangle_{\mathbb{F}_{q^m}},$$

$$\left(\mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},(\boldsymbol{x}_{\check{i}}-x_i)\boldsymbol{y}_{\check{i}})^{\perp}\right)_{\mathbb{F}_{q^m}} = \left\langle \boldsymbol{x}_{\check{i}}^{aq^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{q^\ell};\ a \in [\![0,r-2]\!],\ \ell \in [\![0,m-1]\!] \right\rangle_{\mathbb{F}_{q^m}}.$$

Observe now that

$$\left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)\right)_{\mathbb{F}_{q^m}} = \mathbf{Sh}_i\left(\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)_{\mathbb{F}_{q^m}}\right).$$

Clearly $\boldsymbol{x}^{aq^\ell}(\boldsymbol{x}-x_i)^{q^\ell}\boldsymbol{y}^{q^\ell} = (\boldsymbol{x}^a(\boldsymbol{x}-x_i))^{q^\ell}\boldsymbol{y}^{q^\ell}$ vanishes at i and belongs to $\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)_{\mathbb{F}_{q^m}}$ for a in $[\![0,r-2]\!]$. Therefore $\boldsymbol{x}_{\check{i}}^{aq^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{q^\ell}$ belongs to $\mathbf{Sh}_i\left(\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)_{\mathbb{F}_{q^m}}\right)$. This proves (10). Similarly $y_i^{q^u}\boldsymbol{y}^{q^v} - y_i^{q^v}\boldsymbol{y}^{q^u}$ belongs clearly to $\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)_{\mathbb{F}_{q^m}}$ and vanishes at i. Hence $y_i^{q^u}\boldsymbol{y}_{\check{i}}^{q^v} - y_i^{q^v}\boldsymbol{y}_{\check{i}}^{q^u}$ belongs to $\mathbf{Sh}_i\left(\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)_{\mathbb{F}_{q^m}}\right)$. This proves (12). □

Lemma 22. *Let $\mathscr{B} \stackrel{def}{=} \mathscr{A}_{r-1}(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^\perp$ and $\mathscr{D}' \stackrel{def}{=} \mathscr{B}\star\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)$, then if $r \geqslant q+1$:*

$$\mathscr{B}_{\mathbb{F}_{q^m}} \star \left\langle \boldsymbol{y}_{\check{i}} \right\rangle_{\mathbb{F}_{q^m}} \subseteq \mathscr{D}'.$$

Proof. We use the same notation as in Lemma 21 and observe that $\mathscr{D}_0$ and $\mathscr{D}_1$ defined below are both subcodes of $\mathscr{D}'_{\mathbb{F}_{q^m}}$:

$$\mathscr{D}_0 \stackrel{\text{def}}{=} \mathscr{B}_{\mathbb{F}_{q^m}} \star \mathscr{C}_0 = \left\langle \boldsymbol{x}_{\check{i}}^{aq^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\left(y_i^{q^u}\boldsymbol{y}_{\check{i}}^{q^v+q^\ell} - y_i^{q^v}\boldsymbol{y}_{\check{i}}^{q^u+q^\ell}\right) \mid a\in[\![0,r-2]\!], \ell,u,v\in[\![0,m-1]\!]\right\rangle_{\mathbb{F}_{q^m}}$$

$$\mathscr{D}_1 \stackrel{\text{def}}{=} \mathscr{B}^{\star 2}_{\mathbb{F}_{q^m}} = \left\langle \boldsymbol{x}_{\check{i}}^{aq^j+bq^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^j+q^\ell}\boldsymbol{y}_{\check{i}}^{q^j+q^\ell} \mid a,b\in[\![0,r-2]\!],\ j,\ell\in[\![0,m-1]\!]\right\rangle_{\mathbb{F}_{q^m}}$$

This is a direct consequence of $\mathscr{D}'_{\mathbb{F}_{q^m}} = \mathscr{B}_{\mathbb{F}_{q^m}} \star \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)\right)_{\mathbb{F}_{q^m}}$ (definition of $\mathscr{D}'$ and Point 5 in Lemma 15) and Lemma 21. We also observe that

$$\mathscr{B}_{\mathbb{F}_{q^m}} \star \left\langle \boldsymbol{y}_{\check{i}} \right\rangle_{\mathbb{F}_{q^m}} = \left\langle \left(\boldsymbol{x}_{\check{i}}\right)^{aq^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\left(\boldsymbol{y}_{\check{i}}\right)^{q^\ell+1} \mid a\in[\![0,r-2]\!], l\in[\![0,m-1]\!]\right\rangle_{\mathbb{F}_{q^m}}.$$

Our proof strategy is to start with an element appearing in the vector span above and by suitable additions of elements of $\mathscr{D}_i$ (where $i\in\{0,1\}$), and possibly also by multiplying by some elements in $\mathbb{F}_{q^m}$, results at the end in an element in $\mathscr{D}_i$. This will prove our lemma. We will use here the notation

$$\boldsymbol{u} \stackrel{\mathscr{D}_i}{\to} \boldsymbol{v}$$

to write that $\boldsymbol{v}$ can be obtained from $\boldsymbol{u}$ by adding a suitable element of $\mathscr{D}_i$ and multiplying by some element in $\mathbb{F}_{q^m}$, i.e. this is equivalent to $\boldsymbol{u}-\lambda\boldsymbol{v}\in\mathscr{D}_i$ for a suitable element λ in $\mathbb{F}_{q^m}$. It is readily seen that for any a in $[\![0,r-2]\!]$, u and v in $[\![0,m-1]\!]$ and any polynomial P in $\mathbb{F}_{q^m}[X]$ of degree $\leqslant r-2$ we have

$$P(\boldsymbol{x}_{\check{i}})^{q^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{q^\ell+q^u} \stackrel{\mathscr{D}_0}{\to} P(\boldsymbol{x}_{\check{i}})^{q^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{q^\ell+q^v} \tag{13}$$

$$P(\boldsymbol{x}_{\check{i}})^{q^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{2q^\ell} \stackrel{\mathscr{D}_1}{\to} (\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{2q^\ell}. \tag{14}$$

The first reduction follows by noticing that

$$\begin{aligned} P(\boldsymbol{x}_{\check{i}})^{q^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{q^\ell+q^u} &= P(\boldsymbol{x}_{\check{i}})^{q^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{q^\ell}y_i^{-q^v}\left(y_i^{q^v}\boldsymbol{y}_{\check{i}}^{q^u} - y_i^{q^u}\boldsymbol{y}_{\check{i}}^{q^v} + y_i^{q^u}\boldsymbol{y}_{\check{i}}^{q^v}\right) \\ &= \boldsymbol{d} + y_i^{q^u-q^v}P(\boldsymbol{x}_{\check{i}})^{q^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{q^\ell+q^v} \end{aligned}$$

where $\boldsymbol{d} = P(\boldsymbol{x}_{\check{i}})^{q^\ell}(\boldsymbol{x}_{\check{i}}-x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{q^\ell}y_i^{-q^v}\left(y_i^{q^v}\boldsymbol{y}_{\check{i}}^{q^u} - y_i^{q^u}\boldsymbol{y}_{\check{i}}^{q^v}\right)$ clearly belongs to $\mathscr{D}_0$. The second reduction follows by performing the Euclidean division of $P(X)$ by $(X-x_i)$.

We can namely write $P(X) = (X - x_i)Q(X) + P(x_i)$ for a polynomial Q of degree $\deg P - 1$. Therefore

$$\begin{aligned} P(\boldsymbol{x}_{\check{i}})^{q^\ell}(\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{2q^\ell} &= \left((\boldsymbol{x}_{\check{i}} - x_i)Q(\boldsymbol{x}_{\check{i}}) + P(x_i)\right)^{q^\ell}(\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{2q^\ell} \\ &= \left((\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}Q(\boldsymbol{x}_{\check{i}})^{q^\ell} + P(x_i)^{q^\ell}\right)(\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{2q^\ell} \quad (15) \\ &= \boldsymbol{d} + P(x_i)^{q^\ell}(\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{2q^\ell} \end{aligned}$$

where (15) follows from the $\mathbb{F}_q$-linearity of the Frobenius action $x \mapsto x^{q^\ell}$ and $\boldsymbol{d} = \left((\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}Q(\boldsymbol{x}_{\check{i}})^{q^\ell}\right)(\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{2q^\ell}$ belongs obviously to $\mathscr{D}_1$.

Let us show the inclusion by performing for a generator $\boldsymbol{x}_{\check{i}}^{aq^\ell}(\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{q^\ell+1}$ of $\mathscr{D}_{\mathbb{F}_{q^m}} \star \langle \boldsymbol{y}_{\check{i}} \rangle_{\mathbb{F}_{q^m}}$ a sequence of reductions

$$\begin{aligned} \boldsymbol{x}_{\check{i}}^{aq^\ell}(\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{q^\ell+1} &\xrightarrow{\mathscr{D}_0} \boldsymbol{x}_{\check{i}}^{aq^\ell}(\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{2q^\ell} \\ &\xrightarrow{\mathscr{D}_1} (\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{2q^\ell} \end{aligned}$$

The crucial argument is now the simple observation that

$$(\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell} = \left((\boldsymbol{x}_{\check{i}} - x_i)^q\right)^{q^{\ell^-}}$$

where $\ell^- \stackrel{\text{def}}{=} \ell - 1$ if $\ell > 0$ and $\ell^- \stackrel{\text{def}}{=} m - 1$ if $\ell = 0$. This is a consequence of the fact that the entries of $\boldsymbol{x}$ are in $\mathbb{F}_{q^m}$. This suggests the following sequence of reductions

$$\begin{aligned} (\boldsymbol{x}_{\check{i}} - x_i)^{q^\ell}\boldsymbol{y}_{\check{i}}^{2q^\ell} &\xrightarrow{\mathscr{D}_0} \left((\boldsymbol{x}_{\check{i}} - x_i)^q\right)^{q^{\ell^-}}\boldsymbol{y}_{\check{i}}^{q^\ell+q^{\ell^-}} = \left((\boldsymbol{x}_{\check{i}} - x_i)^{q-1}\right)^{q^{\ell^-}}(\boldsymbol{x}_{\check{i}} - x_i)^{q^{\ell^-}}\boldsymbol{y}_{\check{i}}^{q^{\ell^-}+q^\ell} \\ &\xrightarrow{\mathscr{D}_0} \left((\boldsymbol{x}_{\check{i}} - x_i)^{q-1}\right)^{q^{\ell^-}}(\boldsymbol{x}_{\check{i}} - x_i)^{q^{\ell^-}}\boldsymbol{y}_{\check{i}}^{2q^{\ell^-}} = \left((\boldsymbol{x}_{\check{i}} - x_i)^{q-2}\right)^{q^{\ell^-}}(\boldsymbol{x}_{\check{i}} - x_i)^{2q^{\ell^-}}\boldsymbol{y}_{\check{i}}^{2q^{\ell^-}}. \end{aligned}$$

Note that the last reduction could be performed because the degree of the polynomial $(\boldsymbol{x}_{\check{i}} - x_i)^{q-1}$, which is $q - 1$, is less than or equal to $r - 2$ by assumption on r. For the very same reason ($r \geqslant q + 1$) we observe that the right-hand term $\left((\boldsymbol{x}_{\check{i}} - x_i)^{q-2}\right)^{q^{\ell^-}}(\boldsymbol{x}_{\check{i}} - x_i)^{2q^{\ell^-}}\boldsymbol{y}_{\check{i}}^{2q^{\ell^-}}$ belongs to $\mathscr{D}_1$ which finishes the proof. □

We are ready now to prove Theorem 19.

Proof of Theorem 19. From Lemma 20, we know that $\mathscr{C}_{\mathbb{F}_{q^m}} = \left(\mathscr{A}_r(\boldsymbol{x}_{\check{i}}, \boldsymbol{y}_{\check{i}})^\perp\right)_{\mathbb{F}_{q^m}}$ can be decomposed as

$$\mathscr{C}_{\mathbb{F}_{q^m}} = \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})^\perp\right)\right)_{\mathbb{F}_{q^m}} \oplus \langle \boldsymbol{y}_{\check{i}} \rangle_{\mathbb{F}_{q^m}}.$$

This implies that

$$\begin{aligned} \mathscr{B}_{\mathbb{F}_{q^m}} \star \mathscr{C}_{\mathbb{F}_{q^m}} &= \underbrace{\mathscr{B}_{\mathbb{F}_{q^m}} \star \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})^\perp\right)\right)_{\mathbb{F}_{q^m}}}_{\mathscr{D}'_{\mathbb{F}_{q^m}}} + \mathscr{B}_{\mathbb{F}_{q^m}} \star \langle \boldsymbol{y}_{\check{i}} \rangle_{\mathbb{F}_{q^m}} \\ &= \mathscr{B}_{\mathbb{F}_{q^m}} \star \left(\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})^\perp\right)\right)_{\mathbb{F}_{q^m}} \text{ (by Lemma 22)} \\ &= \mathscr{D}'_{\mathbb{F}_{q^m}}. \end{aligned}$$

The equality of the extended codes over $\mathbb{F}_{q^m}$ implies the equalities of the codes over $\mathbb{F}_q$ which ends the proof. □

3.2. **What is wrong with Goppa codes?** Before moving to the second part of the attack, we make a short digression on how the arguments explained so far (do not) apply to the Goppa case. The discussion below does not represent a proof that computing a filtration is impossible for Goppa codes, but rather an intuition about what hampers it. Goppa codes behave differently from random alternant codes and provide counterexamples to Heuristic 18. The latter should be replaced by

Heuristic 23. *Let $\mathscr{G}(\boldsymbol{x},\Gamma) \stackrel{def}{=} \mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ be a random Goppa code of degree r, with $r \geqslant q-1$ and $\left(\mathscr{G}(\boldsymbol{x},\Gamma)^\perp\right)^{\star 2}$ being different from the full code. Choose an arbitrary code position i and let $\mathscr{C} \stackrel{def}{=} \left(\mathbf{Sh}_i\left(\mathscr{G}(\boldsymbol{x},\Gamma)\right)\right)^\perp$ and $\mathscr{D} \stackrel{def}{=} \mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)^{\star 2}$. Then, with high probability,*

$$\mathbf{Cond}\left(\mathscr{C},\mathscr{D}\right) = \mathscr{A}_r(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^\perp.$$

This is unfortunate, since our approach heavily builds upon the fact that the degree of the conductor decreases. Moreover, it will turn out that the code we obtain as a conductor could have been obtained directly by shortening a suitable code. Actually, it will turn out that for Goppa codes, there are several codes which are very close to each other and which are obtained by various shortenings. This is summarized by the following proposition, whose proof will be given in the appendix. We will explain in what follows why this phenomenon is the main obstacle for applying our conductor approach.

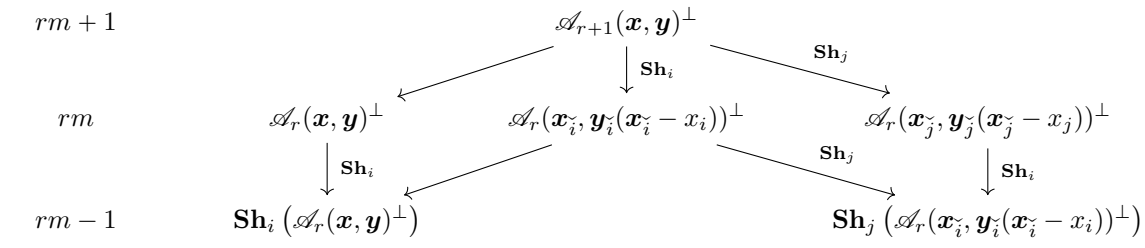
Proposition 24. *Let $\mathscr{G}(\boldsymbol{x},\Gamma) \stackrel{def}{=} \mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ be a Goppa code of degree r. We have for any code positions i and j with $i \neq j$:*

$$\mathscr{A}_{r+1}(\boldsymbol{x},\boldsymbol{y})^\perp = \mathscr{G}(\boldsymbol{x},\Gamma) + \langle\, \mathbf{1} \,\rangle_{\mathbb{F}_q} \quad \textit{(from [Ber00, prop. 1])} \tag{16}$$

$$\mathscr{A}_r(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^\perp = \mathbf{Sh}_i\left(\mathscr{A}_{r+1}(\boldsymbol{x},\boldsymbol{y})^\perp\right) \tag{17}$$

$$\mathbf{Sh}_j\left(\mathscr{A}_r(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^\perp\right) = \mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x}_{\check{j}},\boldsymbol{y}_{\check{j}}(\boldsymbol{x}_{\check{j}}-x_j))^\perp\right). \tag{18}$$

We can summarize these relationships with the diagram below, where arrows mean an inclusion of the lower code into the upper code and two arrows pointing at same code represent the intersection. The typical code dimensions are shown too.



4. Algebraic cryptanalysis

The previous section explains how to obtain, under some conditions, the alternant code $\mathscr{A}_3(\boldsymbol{x}',\boldsymbol{y}')$ with support $\boldsymbol{x}' = \boldsymbol{x}_{\check{\mathcal{I}}}$ and multiplier $\boldsymbol{y}' = \boldsymbol{y}_{\check{\mathcal{I}}}\left(\prod_{i\in\mathcal{I}}(\boldsymbol{x}_{\check{\mathcal{I}}}-x_i)\right)$ for some $\mathcal{I} \subseteq [\![1,n]\!]$ such that $|\mathcal{I}| = r-3$, and with length $n' = n-r+3$ and degree 3, starting from the knowledge of the length-n public code $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$. For the sake

of clarity, in this section we perform algebraic cryptanalysis on the alternant code $\mathscr{A}_3(\boldsymbol{x},\boldsymbol{y})$ of length n. Essentially, we can ignore the structure of $\boldsymbol{y}'$ and the decreased length because the filtration preserves the support and multiplier randomness and the code distinguishability. In Section 4.8, we will see how to get back a support and a multiplier defining $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ (not necessarily $\boldsymbol{x}$ and $\boldsymbol{y}$) from a support and a multiplier defining $\mathscr{A}_3(\boldsymbol{x}',\boldsymbol{y}')$ (not necessarily $\boldsymbol{x}'$ and $\boldsymbol{y}'$). Moreover, we will focus on the case $r=3$ for the system resolution, but the algebraic modeling we describe in Sections 4.1 and 4.2 is more general and makes sense for any $r \geqslant 3$. The full attack needs the filtration to reach degree 3, and therefore works specifically for $q=2$ or $q=3$. However, the specific modeling for $r=3$ described in Section 4.3 is valid for any field size. We describe in Sections 4.4 and 4.5 a polynomial time attack on alternant codes of degree 3 for any $q>2$. This result is original, and to the best of our knowledge, no polynomial time attack was known on non-structured alternant or Goppa codes even for $r=3$. Section 4.6 contains the theoretical proofs and experimental validation for our algorithm.

For $q=2$, we also have a polynomial time attack on random alternant codes of degree 3. However, the algorithm and the theoretical justifications are quite different from the $q>2$ case, they involve algebraic properties of binary alternant codes and are beyond the scope of this article. We just sketch the solving algorithm in Section 4.7. **[mb: cite Rocco's thesis?]**

4.1. **The algebraic modeling from** [FGO$^+$13]**,** $r \geqslant 3$**.** We give an alternative definition of alternant codes which is more suitable for this section. To this end we first need to introduce a rectangular Vandermonde-like matrix:

$$\boldsymbol{V}_r(\boldsymbol{x},\boldsymbol{y}) \stackrel{\text{def}}{=} \begin{bmatrix} y_1 & \dots & y_n \\ y_1x_1 & \dots & y_nx_n \\ \vdots & \ddots & \vdots \\ y_1x_1^{r-1} & \dots & y_nx_n^{r-1} \end{bmatrix},$$

where $\boldsymbol{x}=(x_1,\dots,x_n)\in\mathbb{F}_{q^m}^n$ and $\boldsymbol{y}=(y_1\dots,y_n)\in\mathbb{F}_{q^m}^n$. It is readily seen that an alternant code $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ can be expressed as the code with parity-check matrix $\boldsymbol{V}_r(\boldsymbol{x},\boldsymbol{y})$:

$$\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y}) \stackrel{\text{def}}{=} \left\{\boldsymbol{c}\in\mathbb{F}_q^n \mid \boldsymbol{V}_r(\boldsymbol{x},\boldsymbol{y})\boldsymbol{c}^T=\boldsymbol{0}\right\}$$

We will adopt the notation and follow the description of the algebraic model presented in [FGO$^+$13]. We denote with $\boldsymbol{G}=(g_{i,j})\in\mathbb{F}_q^{k\times n}$ the $k\times n$ generator matrix of $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$. The definition of alternant codes given above implies that

$$\boldsymbol{V}_r(\boldsymbol{x},\boldsymbol{y})\boldsymbol{G}^T=\boldsymbol{0}_{k\times n}.$$

This matrix equation translates into the following system of polynomial equations:

$$\left\{\sum_{j=1}^{n} g_{i,j}Y_jX_j^e=0 \mid i\in[\![1,k]\!], e\in[\![0,r-1]\!]\right\},$$

where $\boldsymbol{X}\stackrel{\text{def}}{=}\{X_1,\dots,X_n\}$ and $\boldsymbol{Y}\stackrel{\text{def}}{=}\{Y_1,\dots,Y_n\}$ are two blocks of n unknowns, each corresponding to the support and multiplier coordinates respectively. Observe that the sought vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ satisfy indeed the polynomial system.

We can assume, up to a permutation of columns, that $\boldsymbol{G}$ is in systematic form, i.e. $\boldsymbol{G}=(\boldsymbol{I}_k \mid \boldsymbol{P})$, where $\boldsymbol{I}_k$ is the identity matrix of size k and $\boldsymbol{P}=(p_{i,j})$ for

$i \in [\![1,k]\!], j \in [\![k+1,n]\!]$. The polynomial system can be therefore rewritten as

$$\left\{ Y_i X_i^e = -\sum_{j=k+1}^{n} p_{i,j} Y_j X_j^e \mid i \in [\![1,k]\!], e \in [\![0,r-1]\!] \right\}. \tag{19}$$

As explained in [FGO+13], thanks to the systematic form assumption, we can get rid of several variables and consider an algebraic system in only $2(n-k)$ unknowns. Indeed it is possible to exploit the trivial identity for $i \in [\![1,k]\!]$

$$Y_i(Y_i X_i^2) = (Y_i X_i)^2$$

and replace the term $Y_i X_i^a$ for $a \in [\![0,2]\!]$ by $-\sum_{j=k+1}^{n} p_{i,j} Y_j X_j^a$ to obtain the equation $\left(\sum_{j=k+1}^{n} p_{i,j} Y_j\right)\left(\sum_{j=k+1}^{n} p_{i,j} Y_j X_j^2\right) = \left(\sum_{j=k+1}^{n} p_{i,j} Y_j X_j\right)^2$. After putting all terms on one side and expanding the products, we obtain that the polynomial

$$f_i \stackrel{\text{def}}{=} \sum_{j,j' \in [\![k+1,n]\!]} p_{i,j} p_{i,j'} Y_j Y_{j'} (X_{j'}^2 - X_j X_{j'})$$

must be zero on the support $\boldsymbol{x}$ and multiplier $\boldsymbol{y}$. Since

$$\begin{aligned} f_i &= \sum_{k+1 \leqslant j < j' \leqslant n} p_{i,j} p_{i,j'} Y_j Y_{j'} (X_j^2 + X_{j'}^2 - 2 X_j X_{j'}) \\ &= \sum_{k+1 \leqslant j < j' \leqslant n} p_{i,j} p_{i,j'} Y_j Y_{j'} (X_j - X_{j'})^2 \end{aligned}$$

we finally obtain the following algebraic system

$$\mathcal{S} \stackrel{\text{def}}{=} \left\{ \sum_{k+1 \leqslant j < j' \leqslant n} p_{i,j} p_{i,j'} Y_j Y_{j'} (X_j - X_{j'})^2 \mid i \in [\![1,k]\!] \right\}. \tag{20}$$

4.2. **Reducing the number of solutions.** The ideal generated by $\mathcal{S}$ is not zero-dimensional. It will be convenient here to reduce to this case by specializing appropriately some variables. The positive dimension of this ideal is due in the first instance to the degrees of freedom for the support and multiplier coordinates. In essence this is due to the fact that a homography $z \mapsto \frac{az+b}{cz+d}$ maps the support $\boldsymbol{x}$ of an alternant code to another support describing the same alternant code (but possibly with a different multiplier) at the condition that $cx_i + d$ never vanishes. When there exists a value x_i of the support of the alternant code for which $cz_i + d = 0$, the resulting code is not an alternant code, but belongs to a slightly larger family of codes : it will be a subfield subcode of a Cauchy code. Let us recall its definition taken from [Dür87]. Given a field $\mathbb{F}$ we can identify the projective line $\mathcal{P}^1(\mathbb{F})$ with $\bar{\mathbb{F}} \stackrel{\text{def}}{=} \mathbb{F} \cup \{\infty\}$, where the symbol ∞ is called *point at infinity*, through the map $\phi : \bar{\mathbb{F}} \to \mathbb{F}^2 \backslash \{0\}, \phi(e) = (e,1)$ if $e \in \mathbb{F}$ and $\phi(\infty) = (1,0)$. Moreover, let $\mathbb{F}[W,Z]_l^H$ be the set of homogeneous polynomials of degree l in two variables W, Z. Given $P \in \mathbb{F}[W,Z]_l^H$ and $e \in \bar{\mathbb{F}}$, we define $P(e) \stackrel{\text{def}}{=} P(\phi(e))$. Then

Definition 25 (Cauchy code)**.** Let $\boldsymbol{x} \stackrel{\text{def}}{=} (x_1, \ldots, x_n) \in \bar{\mathbb{F}}^n$ be a vector of distinct elements and $\boldsymbol{y} \stackrel{\text{def}}{=} (y_1, \ldots, y_n) \in \mathbb{F}^n$ be a vector of nonzero elements. Let $r \in [\![0,n]\!]$. The **Cauchy code** $\mathscr{C}_r(\boldsymbol{x}, \boldsymbol{y})$ is defined as

$$\mathscr{C}_r(\boldsymbol{x}, \boldsymbol{y}) \stackrel{\text{def}}{=} \{(y_1 P(x_1), \ldots, y_n P(x_n)) \mid P \in \mathbb{F}[W,Z]_{r-1}^H\}.$$

As in the case of generalized Reed-Solomon codes, $\boldsymbol{x}$ is called a *support* and $\boldsymbol{y}$ a *multiplier* of the Cauchy code.

If we assume $x_n = \infty$, a generator matrix of $\mathscr{C}_r(\boldsymbol{x}, \boldsymbol{y})$ is given by

$$\begin{bmatrix} y_1 & \cdots & y_{n-1} & 0 \\ y_1 x_1 & \cdots & y_{n-1} x_{n-1} & 0 \\ \vdots & \ddots & \vdots & \\ y_1 x_1^{r-1} & \cdots & y_{n-1} x_{n-1}^{r-1} & y_n \end{bmatrix}. \tag{21}$$

On the other hand, when $\boldsymbol{x} \in \mathbb{F}^n$, *i.e.* when all the x_i's are different from ∞, the Cauchy code $\mathscr{C}_r(\boldsymbol{x}, \boldsymbol{y})$ can be easily seen as $\mathbf{GRS}_r(\boldsymbol{x}, \boldsymbol{y})$. They are also MDS codes. In some sense, Cauchy code are the projective linearization of GRS codes. Analogously subfield subcodes of Cauchy codes generalize subfield subcodes of GRS codes, *i.e.* alternant codes.

One of the main result of [Dür87] was to characterize the possible supports and multipliers of Cauchy codes. In particular, it is proven there that

Theorem 26. [Dür87] *Let $r \in [\![2, n-2]\!]$. Then $\mathscr{C}_r(\boldsymbol{x}, \boldsymbol{y}) = \mathscr{C}_r(\boldsymbol{x}', \boldsymbol{y}')$ if and only if there exists a homography $f(z) = \frac{az+b}{cz+d}$ $(a, b, c, d \in \mathbb{F},\ ad - bc \neq 0)$ such that $\boldsymbol{x}' = f(\boldsymbol{x})$ and $\boldsymbol{y}' = \lambda \theta(\boldsymbol{x})^{r-1} \boldsymbol{y}$ where $\lambda \in \mathbb{F} \backslash \{0\}$ and*

$$\begin{aligned} \theta(z) &= cz + d \quad \text{if } z \in \mathbb{F} \text{ and } cz + d \neq 0, \\ \theta(z) &= (ad - bc)/(-c) \quad \text{if } z \in \mathbb{F} \text{ and } cz + d = 0, \\ \theta(\infty) &= c \quad \text{if } c \neq 0, \\ \theta(\infty) &= a \quad \text{if } c = 0. \end{aligned}$$

Since the elements a, b, c, d in $f(z) = \frac{az+b}{cz+d}$ are defined up to a multiplication by a nonzero scalar, Theorem 26 pragmatically implies that we are allowed to fix three variables in block $\boldsymbol{X}$ and one in block $\boldsymbol{Y}$. The price to pay for this additional specialization is that now we have to eventually handle the point at infinity. We have seen that the column corresponding to the point at infinity in the generator matrix of a Cauchy code has a special form and this changes for this coordinate the form of the system $\mathcal{S}$ given in (20). The problem is that we do not know a priori which x_i will be infinite. To circumvent this problem, we choose the value x_i that will be set to infinity, say x_n.

It is also convenient to make the other choices to belong to the subfield $\mathbb{F}_q$ over which the alternant code is defined: all the Gröbner bases computations will stay in the subfield and this results in slightly improved computation times. Finally we make the following choice (which also simplifies slightly the analysis of the Gröbner basis computations):

$$X_{n-2} = 0, \quad X_{n-1} = 1, \quad X_n = \infty, \quad Y_n = 1, \tag{22}$$

From now on, we denote with a prime the systems where the previous values have been specialized. For instance $\mathcal{S}'$ corresponds to $\mathcal{S}$ with $X_{n-2} = 0, X_{n-1} = 1, X_n = \infty, Y_n = 1$. For the particular specialization $X_n = \infty$, the last column of the Vandermonde matrix is $\begin{pmatrix} 0 & \dots & 0 & 1 \end{pmatrix}^\top$ (see (21)), and the shape of $\mathcal{S}'$ is a little bit different of the one of $\mathcal{S}$ (see next section).

However, the set of solutions of the system $\mathcal{S}'$ still contains a component of positive dimension $n - k - 1 = rm - 1$, that corresponds to the solutions of $\{Y_j =$

$0 \mid k+1 \leqslant j \leqslant n-1\}$. The classical way to deal with the parasite solutions $Y_j = 0$ is to introduce to the system an new variable T_j together with the polynomial $T_jY_j - 1$. This ensures that $Y_j = 0$ is not a solution to the system. However, this also adds variables to the system, and increases the degree of the polynomials during a Gröbner basis computation. A similar phenomenon occurs with the constraints $X_j - X_{j'} \neq 0$. We solve these problems in an easier way in Steps (2) and (3) of the algorithm presented in Section 4.4.

If we substitute (22) into $\mathcal{S}$ for $r \geqslant 4$, we just get the same system as if we had specialized $Y_n = 0$. In the $r = 3$ case it is slightly different.

4.3. Specialized system $\mathcal{S}'$ in the $r = 3$ case.

Proposition 27. *For $r = 3$, we can choose part of the support and multiplier as $X_{n-2} = 0$, $X_{n-1} = 1$, $X_n = \infty$, $Y_n = 1$ and obtain the following algebraic system*

(23)
$$\mathcal{S}' \stackrel{def}{=} \left\{ \sum_{k+1 \leqslant j < j' \leqslant n-1} p_{i,j}p_{i,j'}Y_jY_{j'}(X_j - X_{j'})^2 + \sum_{j=k+1}^{n-1} p_{i,j}p_{i,n}Y_j \mid i \in [\![1,k]\!] \right\}.$$

Proof. The fact that we can choose the support and the multiplier in this way follows from the fact that homographies act 3-transitively on the projective plane and from Theorem 26. To obtain the algebraic system we proceed similarly to what was done to obtain the algebraic system $\mathcal{S}$: this time the Vandermonde matrix is

$$\boldsymbol{V}_3(\boldsymbol{X}, \boldsymbol{Y}) \stackrel{\text{def}}{=} \left[\begin{array}{ccc|ccc} Y_1 & \dots & Y_{n-3} & Y_{n-2} & Y_{n-1} & 0 \\ Y_1X_1 & \dots & Y_{n-3}X_{n-3} & 0 & Y_{n-1} & 0 \\ Y_1X_1^2 & \dots & Y_{n-3}X_{n-3}^2 & 0 & Y_{n-1} & 1 \end{array}\right],$$

so that we get the relations

$$\begin{aligned} Y_i &= \sum_{k+1 \leqslant j \leqslant n-1} p_{i,j}Y_j \\ Y_iX_i &= \sum_{k+1 \leqslant j \leqslant n-1} p_{i,j}Y_jX_j \\ Y_iX_i^2 &= \sum_{k+1 \leqslant j \leqslant n-1} p_{i,j}Y_jX_j^2 + p_{i,n}. \end{aligned}$$

Substituting them into the relations $Y_i(Y_iX_i^2) = (Y_iX_i)^2$ for $i \in [\![1,k]\!]$ gives $\mathcal{S}'$. □

Our purpose is to solve the algebraic system $\mathcal{S}'$ that is given by k polynomials of bidegree $(2,2)$ in the two blocks of variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ and involves $2(n-k)$ variables. The polynomials can be expressed in terms of the $\binom{n-k}{2}$ variables

$$Z_{j,j'} = \begin{cases} Y_jY_{j'}(X_j - X_{j'})^2 & \text{for } k+1 \leqslant j < j' \leqslant n-1, \\ Y_j & \text{for } j' = n, \text{ as } X_n = \infty \text{ and } Y_n = 1. \end{cases} \tag{24}$$

The classical way to solve such system is to first try to linearized the system using the variables $Z_{j,j'}$, for that purpose it is important to know the rank of the associated linearized system.

Its rank is trivially upper bounded by the number of expressions $Z_{j,j'}$, i.e. by $\binom{n-k}{2}$. However, in the high rate regime, [FGO+13] proposed an algebraic heuristic explaining why there is a tighter upper bound. This heuristic was confirmed by intensive computations which showed that the upper-bound was indeed tight. In

[MT21] a proof of the upper bound was given. These results lead us to make the following assumptions, given for the case $r = 3$ of interest, that will be satisfied for a random alternant code.

Assumption 28 (Random alternant code)**.** *We assume that $\mathscr{A}_3(\boldsymbol{x}, \boldsymbol{y})$ is in standard form, and that its dimension satisfies $k = n - rm = n - 3m$.*

Assumption 29 (High rate regime)**.** *If $q \geqslant 3$, we assume that*

$$\mathbf{Rank}(\mathcal{S}) = \mathbf{Rank}(\mathcal{S}') = \binom{3m}{2} - m \leqslant k.$$

Note that for $q = 2$, the rank is smaller:

Assumption 30 (High rate regime)**.** *If $q = 2$, we assume that*

$$\mathbf{Rank}(\mathcal{S}) = \mathbf{Rank}(\mathcal{S}') = \binom{3m}{2} - 3m \leqslant k.$$

This implies that, even after the change of variables $Z_{j,j'}$, the number of unknowns is larger than the number of independent polynomials, and linearization techniques are not enough to solve the system. Therefore, in the following we are going to explain how to tackle this problem with more advanced techniques, namely Gröbner basis.

Note that for any solution $(\boldsymbol{x}, \boldsymbol{y})$ to the system, the vector $(\boldsymbol{x}^q, \boldsymbol{y}^q)$ defined by taking the q-st power coordinate for each coordinate is still a solution to the system. This mean that we can expect at least m solutions.

4.4. **The algorithm for q odd.** Generic Gröbner basis algorithms are not expected to solve efficiently systems with the same degree and same number of unknowns and equations as the one described before. Here however, some expedients can be taken into account to exploit the very strong algebraic structure and specific shape of the equations involved.

We present here an ad hoc algorithm based on Gröbner basis techniques, that recovers the secret key in polynomial time. We start from the system $\mathcal{S}'$ given in (23), under Assumptions 28 and 29, and we agglomerate in our strategy the constraints $Y_j \neq 0$ and $X_j - X_{j'} \neq 0$. This section deals with the odd case (*i.e.* when q is the power of an odd prime), and next one with the even case $q = 2^s$ with $s > 1$. This covers for instance the case $q = 3$, for which a full key recovery can be achieved, thanks to the filtration of alternant codes. The proofs and detailed explanations are posponed in Section 4.6. Our algorithm consists of the following steps:

(1) **Echelonizing system $\mathcal{S}'$** We compute a basis of the $\mathbb{F}_q$-vector space $\mathscr{S}'$ generated by $\mathcal{S}'$. It has the shape

$$\begin{cases} Y_j Y_{j'} (X_j - X_{j'})^2 + L_{j,j'}(Y_\ell : \ell \in I) & \forall k+1 \leqslant j < j' \leqslant n-1 \\ Y_{n-2} Y_{n-1} + L_{n-2,n-1}(Y_\ell : \ell \in I) & \\ Y_j - L_{j,n}(Y_\ell : \ell \in I) & \forall k+1 \leqslant j \leqslant n-1, j \notin I. \end{cases} \tag{25}$$

where $I \subset [\![k+1, n-1]\!]$ has size m, and the $L_{j,j'}$'s are linear functions of the Y_ℓ's, $\ell \in I$. In particular, it contains $2m-1$ homogeneous linear polynomials in Y, that come from the choice $X_n = \infty$ and $Y_n = 1$ (see Proposition 32 for the proof). This can be done in $\mathcal{O}\left(m^{2\omega}\right)$ operations in $\mathbb{F}_q$, where ω is the exponent of linear algebra.

(2) **Removing the $Y_j = 0$ component** For each $j \in [\![k+1, n-1]\!]$, we prove that there exists a set of $2m-1$ linearly independent polynomials in $\mathscr{S}'$ that are multiple of Y_j. As we know that our solution satisfies $Y_j \neq 0$, we add to the system the set $\mathcal{V}_j$ of these polynomials divided by Y_j (see Proposition 33 for details and proof). This has the effect to add $(2m-1)(3m-1)$ linearly independent polynomials of degree 3 to the system, and to remove the non-zerodimensional component from the solution set. Note that Step (1) corresponds to the computation of $\mathcal{V}_n$, as $Y_n = 1$. The cost for all j is $\mathcal{O}\left(m^{\omega+1}\right)$.

(3) **Adding bilinear polynomials** For each $j \in [\![k+1, n-1]\!] \backslash \{n-2\}$, we consider the vector spaces $\mathscr{U}_{j,n-2}$ formed by the polynomials p such that $X_j p \in \mathscr{V}_j + \mathscr{V}_{n-2}$, where $\mathscr{V}_j$ is the $\mathbb{F}_q$-vector space generated by $\mathcal{V}_j$. We prove in Proposition 34 that $\dim_{\mathbb{F}_q}(\mathscr{U}_{j,n-2}) \geqslant m$. Experimentally, this set has dimension exactly m. As we know that our solution satisfies $x_j \neq 0 = x_{n-2}$, we add to the system a basis $\mathcal{U}_{j,n-2}$ of $\mathscr{U}_{j,n-2}$. This has the effect to add (at least) $m(3m-2)$ linearly independent polynomials of degree 2 to the system. The cost for all j is $\mathcal{O}\left(m^{\omega+1}\right)$.

(4) **Eliminating $2m-1$ variables Y_j using the linear polynomials from Step** (1) We now eliminate $2m-1$ variables Y from the polynomials in $\mathcal{U}_{j,n-2}$ using the $2m-1$ homogeneous linear polynomials in Y from Step (1). This step is heuristic, and verified experimentally: the resulting system admits a basis with the shape:

$$\begin{cases} Y_j X_{j'} + f_{j,j'}(Y_\ell : \ell \in I, 1), & \text{for all } j \in I, j' \in [\![k+1, n-3]\!], \\ \sum_{j \in I} a_j Y_j - 1, & \text{with } a_j \in \mathbb{F}_q. \end{cases}$$

where the $f_{j,j'}$ are linear functions of the Y_ℓ's. The basis contains an affine linear polynomial in Y, so that we get in total all $2m$ linearly independent linear polynomials in Y_j that we can expect (see Proposition 31). The cost of this step is $\mathcal{O}\left(m^{2\omega}\right)$.

(5) **Computing linear polynomials in the X variables** Provided that Step (4) occurred as described, we deduce for each j one affine linear polynomial expressing X_j in terms of the Y's (see Proposition 38):

$$\left\{ X_{j'} + \sum_{j \in I} a_j f_{j,j'}(Y_\ell : \ell \in I, 1) \mid j' \in [\![k+1, n-3]\!] \right\}.$$

The cost is $\mathcal{O}\left(m^2\right)$.

(6) **Computing the Gröbner basis** By eliminating the X_i's from the polynomials $Y_j X_{j'} + f_{j,j'}(Y_\ell : \ell \in I, 1)$, we get the final grevlex Gröbner basis of the system, that is experimentally

$$\begin{cases} Y_j Y_{j'} - L'_{j,j'}(Y_i : i \in I_1, 1) & j, j' \in I_1 \\ X_j - L'_{X_j}(Y_i : i \in I_1, 1) & j \in [\![k+1, n-3]\!] \\ Y_j - L'_{Y_j}(Y_i : i \in I_1, 1) & j \notin I_1 \end{cases}$$

for a set $I_1 \subset [\![k+1, n-1]\!]$ of size $m-1$, where the functions L' are affine linear function. This describes a variety of dimension 0 with m solutions, that are exactly the m solutions obtained by applying the Frobenius morphism. The cost is $\mathcal{O}\left(m^{2\omega}\right)$.

(7) **Computing the solutions** The lex basis can be obtained using the FGLM Algorithm from [FGLM93] with $\mathcal{O}\left(m^4\right)$ operations in $\mathbb{F}_q$, and allows to retrieve the m solutions by factorization of a polynomial over $\mathbb{F}_q$ of degree $\leqslant m$.
(8) The final step consists in retrieving separately the values for $Y_1, \ldots, Y_k$ and $X_1, \ldots, X_k$ from (19) for $e = 0$ and $e = 1$. This costs $\mathcal{O}(nm)$ operations in $\mathbb{F}_q$. This needs to be done only once, since with the chosen specialization any of the m solutions is a valid pair of support and multiplier coordinates, thus we can choose arbitrarily any of them.

Note that all steps are just linear algebra, and the total complexity is polynomial in m and n, the global cost being $\mathcal{O}\left(m^{2\omega} + nm\right)$ as $m, n \to \infty$.

4.5. **The algorithm for q even, $q > 2$.** We present here the differences for the case of characteristic 2. We still assume Assumptions 28 and 29. Note that the case $q = 2$ is very different, we will consider this case later in Section 4.7. We assume here that $q = 2^s$ with $s > 1$.

First of all, in this case the system $\mathcal{S}'$ can be rewritten as

$$\mathcal{S}' = \left\{ \sum_{k+1 \leqslant j < j' \leqslant n-1} p_{i,j} p_{i,j'} Y_j Y_{j'} (X_j^2 + X_{j'}^2) + \sum_{j=k+1}^{n-1} p_{i,j} p_{i,n} Y_j \mid i \in [\![1, k]\!] \right\}.$$

Since the X_j's variables appear in the system with power 2 only, we can perform a change of variables by defining $W_j \stackrel{\text{def}}{=} X_j^2$, so that the system becomes

$$\mathcal{S}'_2 = \left\{ \sum_{k+1 \leqslant j < j' \leqslant n-1} p_{i,j} p_{i,j'} Y_j Y_{j'} (W_j + W_{j'}) + \sum_{j=k+1}^{n-1} p_{i,j} p_{i,n} Y_j \mid i \in [\![1, k]\!] \right\}. \tag{26}$$

Therefore, polynomials in $\mathcal{S}'_2$ have bidegree $(1, 2)$ in $\boldsymbol{W} \stackrel{\text{def}}{=} \{W_1, \ldots, W_n\}$ and $\boldsymbol{Y}$ respectively. This simple trick provides an effective speed up to the resolution.

Steps (1)–(2) are identical, we compute for each $j \in [\![k+1, n-1]\!]$ the vector space $\mathscr{V}_j$ that has dimension $2m-1$. The difference is that the polynomials in $\mathcal{V}_j$ for $j \leqslant n-1$ are linear combination of the monomials $Y_{j'}(W_j + W_{j'})$, $j' \in [\![k+1, n-1]\!]$, $j' \neq j$, and 1, hence already bilinear. Step (3) is unnecessary.

Step (4): using the linear polynomials in $\mathcal{V}_n$, we can directly eliminate $2m-1$ variables Y_j from the $(3m-1)(2m-1)$ bilinear polynomials in $\cup_{j=k+1}^{n-1} \mathcal{V}_j$. We obtain experimentally a basis of the shape

$$\begin{cases} Y_j W_{j'} + g_{j,j'}(Y_\ell : \ell \in I, 1), & \text{for all } j \in I, j' \in [\![k+1, n-3]\!], \\ \sum_{j \in I} a_j Y_j - 1, & \text{with } a_j \in \mathbb{F}_q. \end{cases} \tag{27}$$

Steps (5)–(8) are identical to the odd case. Remark that retrieving the value x_j from the values w_j is just a square root computation in characteristic 2.

The final asymptotic cost is the same, $\mathcal{O}\left(m^{2\omega} + nm\right)$ as $n, m \to \infty$.

4.6. **Theoretical and experimental validation of the algebraic algorithm.** We start with a property that will be useful to determine the number of linearly independent linear polynomials in $\mathscr{S}'$, for any q and any $r \geqslant 3$. Recall that we assumed without loss of generality that $y_n = 1$.

Proposition 31. *Let $\mathscr{C}$ be the $\mathbb{F}_{q^m}$ linear code generated by $(y_{k+1},\dots,y_{n-1})$ in $\mathbb{F}_{q^m}^{mr-1}$. Then, under Assumption 28, we have*

$$\dim_{\mathbb{F}_q}(Tr(\mathscr{C})) = m,$$
$$\dim_{\mathbb{F}_q}(Tr(\mathscr{C})^\perp) = (r-1)m-1.$$

As any $\mathbb{F}_q$-linear combination of the y_j's that is equal to zero provides a codeword in $Tr(\mathscr{C})^\perp$, therefore there cannot be more than $(r-1)m-1$ linearly independent homogeneous $\mathbb{F}_q$-linear polynomials in $Y_{k+1},\dots,Y_{n-1}$ which cancel on $y_{k+1},\dots,y_{n-1}$, and no more than $(r-1)m$ linearly independent affine $\mathbb{F}_q$-linear polynomials in $Y_{k+1},\dots,Y_{n-1}$ that cancel on $(y_{k+1},\dots,y_{n-1},1)$.

Equivalently, for all $j \in [\![k+1, n-1]\!]$, the code $\mathscr{C}'_j \subset \mathbb{F}_{q^m}^{mr-1}$ generated by $(y_{k+1}(x_{k+1}-x_j)^{r-1},\dots,y_{n-1}(x_{n-1}-x_j)^{r-1},1) \in \mathbb{F}_{q^m}^{mr}$ and punctured in position $j-k$ satisfies

$$\dim_{\mathbb{F}_q}(Tr(\mathscr{C}'_j)) = m,$$
$$\dim_{\mathbb{F}_q}(Tr(\mathscr{C}'_j)^\perp) = (r-1)m-1.$$

Proof. If the code $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ has dimension $k = n-mr$ and is in standard form, then the last $n-k = mr$ columns of its parity-check matrix must be an information set, i.e. the last mr columns of $\boldsymbol{V}_r(\boldsymbol{x},\boldsymbol{y})$ must generate a trace code with dimension mr. This means in particular that the first row $(y_{k+1},\dots,y_{n-1},0)$ must have rank weight m, and this is the same for all r rows. Then, by elementary combination of rows, the trace code of the following code must still have dimension m:

$$\begin{pmatrix} 0 & y_{k+2}(x_{k+2}-x_{k+1})^{r-1} & \dots & y_{n-1}(x_{n-1}-x_{k+1})^{r-1} & 1 \end{pmatrix}$$

and this can be done for any x_j instead of x_{k+1}, hence the proposition. □

We now give details for our algorithm for $r = 3, q \geqslant 3$.

Step (1)*: Echelonizing system $\mathcal{S}'$.* We linearize the set of polynomials (23), by replacing "polynomials" by variables, instead of classically replacing any monomial by a new variable. The variables we consider are:

$$Z_{j,j'} = \begin{cases} Y_jY_{j'}(X_j-X_{j'})^2 & \text{for } k+1 \leqslant j < j' \leqslant n-1 \\ Y_j & \text{for } j \in [\![k+1, n-1]\!], j' = n. \end{cases}$$

Proposition 32. *Assume that $q \geqslant 3$. Under Assumptions 28 and 29, for any set $I \subset [\![k+1, n-1]\!]$ of size m such that $\dim_{\mathbb{F}_q}(\langle y_\ell : \ell \in I\rangle_{\mathbb{F}_q}) = m$, a basis of $\mathscr{S}'$ is given by*

$$\begin{cases} Y_jY_{j'}(X_j-X_{j'})^2 + L_{j,j'}(Y_\ell : \ell \in I) & \forall k+1 \leqslant j < j' \leqslant n-1 \\ Y_{n-2}Y_{n-1} + L_{n-2,n-1}(Y_\ell : \ell \in I) & \\ Y_j - L_{j,n}(Y_\ell : \ell \in I) & \forall k+1 \leqslant j \leqslant n-1, j \notin I. \end{cases}$$

where the $L_{j,j'}$ are linear functions of the Y_ℓ's, $\ell \in I$ (note that $L_{j,j'}$ implicitly depends on I). This basis can be computed in time $\mathcal{O}\left(m^{2\omega}\right)$ where ω is the constant of linear algebra.

For q even, we get the same basis for $\mathcal{S}'_2$ with the terms $(X_j-X_{j'})^2$ replaced by $W_j+W_{j'}$.

Proof. We have $\binom{3m}{2} - m$ polynomials in $\binom{3m}{2}$ variables. Among the variables, $3m-1$ are of degree 1 (the Y_j's for $k+1 \leqslant j \leqslant n-1$), one is of degree 2 ($Z_{n-2,n-1} = -Y_{n-2}Y_{n-1}$, as $X_{n-2} = 0$ and $X_{n-1} = 1$) and the last $\binom{3m}{2} - 3m$ are of degree 4. We can eliminate from the system all terms of degree 4 and 2. As the polynomials are linearly independent, we get at least $2m-1$ linear polynomials in the Y_j's.

As by Proposition 31, we have at most $2m-1$ linear relations between the $Y_{k+1}, \ldots, Y_{n-1}$, hence we have exactly $2m-1$ linear polynomials in the Y_j's expressing any Y_j in terms of the $\{Y_\ell : \ell \in I\}$ for some $I \subset [\![k+1, n-1]\!]$ of size m, and all other polynomials express the terms of degree $\geqslant 2$ in terms of the $\{Y_\ell : \ell \in I\}$.

To compute the basis it is enough to compute an echelon form of a matrix of size $(\binom{3m}{2} - m) \times \binom{3m}{2}$, the cost is $\mathcal{O}\left(m^{2\omega}\right)$. □

Step (2)*: removing the* $Y_j = 0$ *component.* The linear polynomials we get come from the fact that we have specialized the nth component to $x_n = \infty$ and $y_n = 1$. Here we show that it is equivalently possible to introduce the constraint $Y_j \neq 0$ for all $j \in [\![k+1, n-1]\!]$. Let $\boldsymbol{X} = (X_{k+1}, \ldots, X_{n-3})$ and $\boldsymbol{Y} = (Y_{k+1}, \ldots, Y_{n-1})$. We define the vector spaces

$$\mathscr{V}_j = \frac{1}{Y_j}\left(\mathscr{S}' \cap Y_j \cdot \mathbb{F}_q[\boldsymbol{X}, \boldsymbol{Y}]_{\leqslant 3}\right), \qquad j \in [\![k+1, n-1]\!] \tag{28}$$

that is $\mathscr{V}_j \stackrel{\text{def}}{=} \langle \frac{h_1}{Y_j}, \ldots, \frac{h_\ell}{Y_j} \rangle_{\mathbb{F}_q}$ where $\{h_1, \ldots, h_\ell\}$ is a basis of $\mathscr{S}' \cap (Y_j \cdot \mathbb{F}_q[\boldsymbol{X}, \boldsymbol{Y}]_{\leqslant 3})$. We also define

$$\mathscr{V}_n = \langle Y_j - L_{j,n}(Y_\ell, \ell \in I) \rangle_{j \in \{k+1..n-1\}, j \notin I}. \tag{29}$$

Proposition 33. *Under Assumptions* (28) *and* (29)*, for any* $q \geqslant 3$*,*

$$\dim_{\mathbb{F}_q}(\mathscr{V}_j) = 2m-1 \qquad \forall k+1 \leqslant j \leqslant n \tag{30}$$

and any polynomial in $\mathscr{V}_j$ *is a linear combination of the* $3m-1$ *terms*

$$\begin{cases} Y_{j'}(X_j - X_{j'})^2, & j' \in [\![k+1, n-1]\!], j' \neq j \\ 1 \end{cases}$$

We also have, for each $j_1, j_2 \in [\![k+1, n-1]\!]$ *with* $j_1 \neq j_2$*:*

$$\dim_{\mathbb{F}_q}(\mathscr{V}_{j_1} + \mathscr{V}_{j_2}) = 4m-2.$$

A basis $\mathcal{V}_j$ *of* $\mathscr{V}_j$ *can be computed in time* $\mathcal{O}\left(m^\omega\right)$ *from the basis* (25) *of* $\mathscr{S}'$ *and the set of all* $\mathcal{V}_j$*'s can be computed in time* $\mathcal{O}\left(m^{\omega+1}\right)$*.*

Proof. Choose any set $I \subset [\![k+1, n-1]\!]$ of size m such that $\dim_{\mathbb{F}_q}\left(\left\langle (y_\ell)_{\ell \in I} \right\rangle_{\mathbb{F}_{q^m}}\right) = m$.

a) Consider first the case where $j \notin I$, and $j \leqslant n-3$. To compute $\mathscr{V}_j$, we just take the polynomials in (25) that contain Y_j, they are the $3m-1$ polynomials:

$$\begin{cases} Y_j Y_{j'}(X_j - X_{j'})^2 + L_{j,j'}(Y_\ell : \ell \in I) & \text{for } j' \in [\![k+1, n-1]\!], j' \neq j \\ Y_j - L_{j,n}(Y_\ell : \ell \in I). \end{cases}$$

We have $3m-1$ linearly independent polynomials in m variables $Y_\ell : \ell \in I$ and $3m-1$ variables $Y_j Y_{j'}(X_j - X_{j'})^2$ and Y_j. By eliminating the $Y_\ell : \ell \in I$ we get at least $2m-1$ polynomials that are multiple of Y_j.

b) If $j \in I$, $j \leqslant n-3$, we have $\dim_{\mathbb{F}_q}\left(\left\langle (y_i)_{i\in[\![k+1,n-1]\!]\setminus\{j\}}\right\rangle_{\mathbb{F}_{q^m}}\right) \in \{m-1,m\}$. If the dimension over $\mathbb{F}_q$ is m, then we can take a different set I that generates a $\mathbb{F}_q$-vector space $\left\langle (y_i)_{i\in I}\right\rangle_{\mathbb{F}_{q^m}}$ of dimension m such that $j \notin I$. If the dimension over $\mathbb{F}_q$ is $m-1$, then for any $j' \in [\![k+1,n-1]\!]\setminus I$, the linear polynomials $Y_{j'} - L_{j',n}(Y_\ell : \ell \in I)$ does not involve Y_j (or we would have a linear polynomial expressing Y_j as a $\mathbb{F}_q$-linear combination of the terms $Y_{j'}$ and $Y_\ell : \ell \in I\setminus\{j\}$, which is impossible considering that $\dim_{\mathbb{F}_q}\left(\left\langle (y_i)_{i\in[\![k+1,n-1]\!]}\right\rangle_{\mathbb{F}_{q^m}}\right) = m$). In this case, we take the $3m-2$ polynomials involving $Y_jY_{j'}(X_j - X_{j'})^2$ for all $j' \neq j$, they contains those $3m-2$ terms, the variable Y_j and $m-1$ variables $Y_\ell : \ell \in I, \ell \neq j$. By eliminating the $Y_\ell, \ell \in I, \ell \neq j$ we get at least $2m-1$ linear polynomials multiple of Y_j.

c) For $j \in [\![n-2,n-1]\!]$, it is exactly the same as a) and b), but with one more polynomial involving one more variable $Y_{n-2}Y_{n-1}$.

In any case, after dividing by Y_j, we get at least $2m-1$ polynomials involving the monomials $Y_{j'}(X_{j'} - X_j)^2$ for $j' \in [\![k+1,n-1]\!]\setminus\{j\}$ and 1, and all those polynomials evaluate to zero on the support and multiplier of the code. Now, according to Proposition 31 with $r=3$, $\dim_{\mathbb{F}_q}(\mathrm{Tr}\left(\mathscr{C}'_j\right)) = m$, where $\mathscr{C}'_j$ is the code of length $3m-1$ generated by $(y_{k+1}(x_{k+1}-x_j)^2, \ldots, y_{n-1}(x_{n-1}-x_j)^2, 1)$ and punctured in position $j-k$ (where its value is 0). Hence there can be at most $2m-1$ linear polynomials between the $3m-1$ terms 1 and $Y_{j'}(X_{j'}-X_j)^2$, $j' \in [\![k+1,n-1]\!], j' \neq j$.

Finally, the polynomials in two different $\mathscr{V}_j$'s are linearly independent, as the only common polynomial that could belong simultaneously to two different $\mathscr{V}_j$'s is 1, and the ideal is not generated by 1. □

Step (3)*: adding bilinear polynomials.* The system given by the union of $\mathcal{S}'$ and the $(3m-1)(2m-1)$ cubic polynomials $\mathcal{V}_j, j \in [\![k+1,n-1]\!]$ determined at Step (2) generates a zero-dimensional ideal, whose variety contains exactly m solutions. It would be enough to run a Gröbner basis for this new system, in order to retrieve the support and multiplier. However, specifically for the q odd case, we are able to deepen the analysis and use efficiently the constraints about support coordinates, *i.e.* $X_{j_1} - X_{j_2} \neq 0$, by computing efficiently a set of bilinear polynomials. The latter do not refine the variety, this one being already finite, but their prediction allows to speed up the computation.

Proposition 34. *The vector space* $\mathscr{U}_{n-2,n-1} \stackrel{def}{=} (\mathscr{V}_{n-2} + \mathscr{V}_{n-1}) \cap \mathbb{F}_q[\boldsymbol{X},\boldsymbol{Y}]_{\leqslant 2}$ *contains more than* m *linearly independent polynomials of degree* 2*, that are linear combination of the terms* $Y_j(2X_j-1)$ *for* $j \in [\![k+1,n-1]\!]$*. We denote by* $\mathcal{U}_{n-2,n-1}$ *this set of* m *polynomials.*

Moreover, $\mathscr{V}_{n-2} + \mathscr{V}_{n-1}$ *contains an additional polynomial of degree* 2 *expressing the monomial 1 in terms of the* $Y_j(2X_j-1)$ *for* $j \in [\![k+1,n-1]\!]$*, denoted by* $u_{n-2,n-1}$*.*

If the characteristic of the field is 2*, the* $m+1$ *polynomials are linear in* Y_j *and* 1*. The first* m *polynomials in* $U_{n-2,n-1}$ *already belong to* $\mathscr{V}_n$*, but the additional polynomial* $u_{n-2,n-1}$ *generates together with* $\mathcal{V}_n$ *a vector space of dimension* $2m$ *(hence one more linear polynomial than the ones in* $\mathcal{V}_n$*).*

Proof. The terms appearing in the polynomials in $\mathcal{V}_{n-2}$ are $1, Y_{n-1}$ and $Y_jX_j^2$ for $j \in [\![k+1,n-3]\!]$. The ones in $\mathcal{V}_{n-1}$ are $1, Y_{n-2}$ and $Y_jX_j^2 + Y_j(1-2X_j)$, $j \in [\![k+1,n-3]\!]$. This means that the polynomials in $\mathscr{U}_{n-2,n-1}$ can all be expressed as linear combination of the $3m-3$ monomials $Y_jX_j^2$ of degree 3, the monomial

1 of degree 0, plus $3m-1$ terms of degree at most 2: the monomials Y_{n-1}, Y_{n-2} and the $Y_j(2X_j-1)$'s, $j \in [\![k+1, n-3]\!]$. The dimension of the vector space $\mathscr{U}_{n-2,n-1}$ is $4m-2$, so that we get at least m linearly independent polynomials of degree 2 in $\mathscr{U}_{n-2,n-1}$ that are combination of $3m-1$ terms $(Y_{k+1}(2X_{k+1}-1), \ldots, Y_{n-3}(2X_{n-3}-1), Y_{n-2}, Y_{n-1})$. If the characteristic of the field is 2, then we get m linear polynomials in Y, that are already in $\mathscr{V}_n$ according to Proposition 31.

In all cases, we also get an additional polynomial of degree 2 involving these $3m-1$ terms and the monomial 1, and this gives in characteristic 2 an additional affine linear polynomial in Y. □

This can be generalized to the following vector spaces. For any $k+1 \leqslant j_1 < j_2 \leqslant n-1$, define the vector space

$$\mathscr{U}_{j_1,j_2} = \frac{1}{X_{j_1}-X_{j_2}} \left((\mathscr{V}_{j_1} + \mathscr{V}_{j_2}) \cap (X_{j_1}-X_{j_2}) \cdot \mathbb{F}_q[\boldsymbol{X}, \boldsymbol{Y}]_{\leqslant 2} \right) \tag{31}$$

that consists of the polynomials p such that $p(X_{j_1}-X_{j_2}) \in \mathscr{V}_{j_1} + \mathscr{V}_{j_2}$.

Proposition 35. *For any $k+1 \leqslant j_1 < j_2 \leqslant n-1$, $(j_1, j_2) \neq (n-2, n-1)$, we have $\dim(\mathscr{U}_{j_1,j_2}) \geqslant m$, and the polynomials in $\mathscr{U}_{j_1,j_2}$ are linear combination of the following terms:*

$$Y_j(2X_j - X_{j_1} - X_{j_2}), \qquad\qquad j \in [\![k+1, n-1]\!].$$

Proof. The $2m-1$ polynomials in $\mathcal{V}_{j_1}$ contains the following $3m-1$ terms:

$$\begin{cases} Y_j(X_j - X_{j_1})^2 & j \in [\![k+1, n-1]\!], j \neq j_1 \\ 1 \end{cases}$$

It is the same for $\mathcal{V}_{j_2}$, but we can rewrite, for $j \in [\![k+1, n-1]\!], j \neq j_1, j_2$:

$$Y_j(X_j - X_{j_2})^2 = Y_j(X_j - X_{j_1})^2 + Y_j(2X_j - X_{j_1} - X_{j_2})(X_{j_1} - X_{j_2}),$$

so that the $4m-2$ polynomials generating $\mathscr{V}_{j_1} + \mathscr{V}_{j_2}$ can be written in terms of the following terms:

$$\begin{cases} Y_j(X_j - X_{j_1})^2 & j \in [\![k+1, n-1]\!], j \neq j_1, j_2, \\ 1 & \\ Y_{j_1}(X_{j_1} - X_{j_2})^2 & \\ Y_{j_2}(X_{j_1} - X_{j_2})^2 & \\ Y_j(2X_j - X_{j_1} - X_{j_2})(X_{j_1} - X_{j_2}), & j \in [\![k+1, n-1]\!], j \neq j_1, j_2. \end{cases}$$

If we eliminate the $3m-2$ first terms that are not multiple of $X_{j_1} - X_{j_2}$, we get at least m linearly independent polynomials that are multiple of $X_{j_1} - X_{j_2}$. After division by $X_{j_1} - X_{j_2}$, the polynomials are linear combination of the $3m-1$ terms $Y_j(2X_j - X_{j_1} - X_{j_2})$ for $j \in [\![k+1, n-1]\!]$.

□

Remark 36. If the characteristic of the field is 2, then we can divide directly by $X_{j_1}^2 + X_{j_2}^2$ to get m linear polynomials in the Y_j's. As we cannot have more than $2m-1$ homogeneous linear polynomials in the Y_j's if the multipliers y_j generate $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$, then all the new polynomials are linearly dependent from the previous ones.

Despite the existence of a quadratic number of vector spaces $\mathcal{U}_{j_1,j_2}$, in practice it is sufficient to exploit the polynomials derived from the $3m-2$ subspaces $\mathcal{U}_{\ell,n-2},\ \ell \in [\![k+1, n-1]\!] \backslash \{n-2\}$, thus reducing the complexity of this step.

Fact 37. Experimentally, $\dim(\mathscr{U}_{j_1,j_2}) = m$ and for odd q,

$$\dim_{\mathbb{F}_q}(\cup_{\ell \in [\![k+1,n-1]\!], \ell \neq n-2} \mathscr{U}_{\ell,n-2}) = m(3m-2).$$

Step (4)*: eliminating* $2m-1$ *variables* Y_j *using the linear polynomials from Step* (1). We consider the case q odd. Assuming Fact 37, the system $\oplus_{j_1 \in [\![k+1,n-3]\!]} \mathcal{U}_{j_1,n-2}$ contains $m(3m-2)$ linearly independent polynomials, and they can all be expressed as linear combination of the monomials $Y_j, j \in [\![k+1, n-1]\!]$ and $Y_j X_{j'}$ for $j \in [\![k+1, n-1]\!]$ and $j' \in [\![k+1, n-3]\!]$.

The system $\mathcal{V}_n$ contains $2m-1$ homogeneous linear polynomials in the Y_j's, expressing the Y_i's for $i \notin I$ in term of the Y_ℓ's for $\ell \in I$. If we use them to eliminate the $2m-1$ variables Y_i's ($i \notin I$) from the polynomials in $\oplus_{j_1 \in [\![k+1,n-3]\!]} \mathcal{U}_{j_1,n-2}$, we are left with polynomials that are linear combination of m linear monomials $\{Y_\ell : \ell \in I\}$, and $m(3m-3)$ quadratic monomials $Y_j X_{j'}$ for $j \in I$ and $j' \in [\![k+1, n-3]\!]$. This means that we have as many polynomials as monomials. However, the polynomials now have no reason to remain linearly independent, and in fact they are not.

Experimentally, after linearization, we get one polynomial expressing each quadratic term $Y_j X_{j'}$ in terms of the m independent $\{Y_\ell : \ell \in I\}$ and m reductions to zero, as we cannot get more than $2m-1$ linear polynomials relating the Y_j's: a basis $\mathcal{U}$ of $\oplus_{j_1} \mathscr{U}_{j_1,n-2}$ modulo $\mathcal{V}_n$ has the shape

$$\mathcal{U} \stackrel{\text{def}}{=} \left\{ Y_j X_{j'} + f_{j,j'}(Y_\ell : \ell \in I) \mid j \in I, j' \in [\![k+1, n-3]\!] \right\}.$$

We can now use the polynomial $u_{n-2,n-1}$ from Proposition 34, that is a linear combination of the monomials $1, Y_{n-2}, Y_{n-1}$ and the $Y_j(2X_j - 1)$ for $j \in [\![k+1, n-3]\!]$. We eliminate the Y_j's, $j \notin I$ using equations in $\mathcal{V}_n$ and the $Y_j X_{j'}$ for $j \in I, j' \in [\![k+1, n-3]\!]$ using $\mathcal{U}$ and obtain a linear polynomials in the Y_j's and 1. Note that, as we already have $2m-1$ homogeneous polynomials between the Y_j's, we cannot have another homogeneous polynomials, hence the polynomials contains the constant 1.

To perform the elimination and the linearization, we can perform linear algebra on a matrix where the columns are the $Y_j X_{j'}$, hence $\mathcal{O}\left(m^2\right)$ columns, and the rows are the basis for $\mathscr{U}_{j,n-2}$ and the $X_i L_j$ with L_j a linear polynomial in $\mathscr{V}_n$. This makes $\mathcal{O}\left(m^2\right)$ rows, and a complexity in $\mathcal{O}\left(m^{2\omega}\right)$.

Step (5)*: computing linear polynomials for the* X *variables.*

Proposition 38. *Assume that a basis of* $\cup_{j \neq n-2} \mathscr{U}_{j,n-2}$ *where the linear polynomials from* $\mathcal{V}_n$ *have been eliminated is given by*

$$\begin{cases} Y_j X_{j'} + f_{j,j'}(Y_\ell : \ell \in I, 1), & \text{for all } j \in I, j' \in [\![k+1, n-3]\!], \\ \sum_{j \in I} a_j Y_j - 1, & \text{with } a_j \in \mathbb{F}_q. \end{cases} \tag{32}$$

Then the vector space generated by the polynomials (32) *contains the polynomials*

$$X_{j'} + \sum_{j \in I} a_j f_{j,j'}(Y_\ell : \ell \in I, 1), \qquad j' \in [\![k+1, n-3]\!]. \tag{33}$$

Proof. We have

$$\sum_{j\in I} a_j(Y_jX_{j'} + f_{j,j'}(Y_\ell : \ell \in I, 1)) = X_{j'} + X_{j'}(\sum_{j\in I} a_jY_j - 1) + \sum_{j\in I} a_j f_{j,j'}(Y_\ell : \ell \in I, 1)$$

so that we get in the ideal generated by (32) one affine linear polynomial expressing each $X_{j'}$ in terms of the $\{Y_\ell : \ell \in I, 1\}$. □

Step (6)*: the final Gröbner basis.* Now, if we use the polynomials in (33) to eliminate the X_i's from the polynomials in (32), we get one linear polynomial for each term of degree 2 in $\boldsymbol{Y}$. Let I_1 be the set I minus one element $i \in I$ such that $a_i \neq 0$. The final basis has the shape

$$\begin{cases} Y_jY_{j'} + L'_{j,j'}(Y_\ell : \ell \in I_1, 1), & j, j' \in I_1, j < j', \\ X_{j'} + f_{j'}(Y_\ell : \ell \in I_1, 1), & j' \in [\![k+1, n-3]\!], \\ Y_j + L_{j,n}(Y_\ell : \ell \in I_1), & j \notin I, \\ Y_i + L_{i,n}(Y_\ell : \ell \in I_1, 1), & i \in I \backslash I_1. \end{cases}$$

This describes a variety of dimension 0 with m solutions (as $\#I_1 = m-1$, only m monomials 1 and $Y_i : i \in I_1$ are not leading term of a polynomial in the ideal), that are exactly the m solutions obtained by applying the Frobenius morphism. This proves that the basis is a Gröbner basis. It coincides with the basis that would have been computed from $\mathscr{S}'$ plus the cubic polynomials from the $\mathcal{V}_j$'s. However, our approach is more efficient, because it avoids unnecessary calculations.

4.7. **The particular $q = 2$ case.** Assumption 30 asserts that the rank of $\mathcal{S}'$ is smaller than for all the other cases, namely $\mathbf{Rank}(\mathcal{S}') = \binom{3m}{2} - 3m$ instead of $\binom{3m}{2} - m$. This invalidates all the combinatorial arguments for the dimensions of $\mathcal{V}_j$'s and for the number of degree falls. In this case we have

$$\dim(\mathcal{V}_j) = m - 1,$$

In particular for $j = n$, the number of independent linear polynomials in $\boldsymbol{Y}$ in a basis of $\mathscr{S}'$ is $m - 1$. This result is rather technical, its essence not being merely combinatorial, and is beyond the scope of this article.

More interestingly, for the specific $q = 2$ case, we can use the following particular identity from [FGO$^+$13]:

$$(Y_iX_i^2)(Y_i)^2 = (Y_i)(Y_iX_i)^2. \tag{34}$$

Replacing as before the terms $Y_iX_i^a$ for $a \in [\![0, 2]\!]$ by their values from (19) and, as $q = 2$, the terms $(Y_iX_i^a)^2$ for $a \in [\![0, 1]\!]$ by $(Y_iX_i^a)^2 = -\sum_{j=k+1}^{n} p_{i,j}Y_j^2X_j^{2a}$ leads, after expansion, to the system

$$\mathcal{B} \stackrel{\text{def}}{=} \left\{ \sum_{k+1\leqslant j<j'\leqslant n} p_{i,j}p_{i,j'}Y_jY_{j'}(Y_j + Y_{j'})(X_j + X_{j'})^2 = 0 \mid i \in [\![1, k]\!] \right\}. \tag{35}$$

Proposition 39. *For $r = 3$, $q = 2$ with the specialization* (22)*, the system $\mathcal{B}$ becomes*

(36)
$$\mathcal{B}' = \left\{ \sum_{\substack{j,j'=k+1 \\ j<j'}}^{n-1} p_{i,j}p_{i,j'}Y_jY_{j'}(Y_j+Y_{j'})(X_j+X_{j'})^2 + \sum_{j=k+1}^{n-1} p_{i,j}p_{i,n}Y_j^2 \mid i \in [\![1,k]\!] \right\}.$$

Proof. For $q = 2$ we use the identity $(Y_iX_i^2)(Y_i)^2 = (Y_i)(Y_iX_i)^2$, and the fact that for $q = 2$, $Y_i^2 = \sum_{k+1\leqslant j\leqslant n-1} p_{i,j}Y_j^2$ and $(Y_iX_i)^2 = \sum_{k+1\leqslant j\leqslant n-1} p_{i,j}Y_j^2X_j^2$. After expansion, using the fact that $Y_jX_{j'}^2+Y_{j'}X_{j'}^2+Y_{j'}X_j^2+Y_jX_j^2 = (Y_j+Y_{j'})(X_j^2+X_{j'}^2)$ for any $k+1 \leqslant j < j' \leqslant n-1$, we get $\mathcal{B}'$. □

As before, we call $\mathscr{B}'$ the vector space generated by the polynomials $\mathcal{B}'$. The system $\mathcal{B}'$ contains $\binom{3m}{2} - 3m$ linearly independent polynomials of bidegree $(2,3)$ in the two blocks of variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ and its linearization involves $\binom{3m}{2}$ variables

$$\begin{cases} Y_jY_{j'}(Y_j+Y_{j'})(X_j^2+X_{j'}^2) & \text{for } k+1 \leqslant j < j' \leqslant n-1, \\ Y_j^2 & \text{for } j' = n \text{ and } X_n = \infty. \end{cases} \tag{37}$$

In a similar way to what done for $\mathcal{S}'$, we can define the vector spaces

$$\mathscr{E}_j = \frac{1}{Y_j}\left(\mathscr{B}' \cap Y_j \cdot \mathbb{F}_q[\boldsymbol{X},\boldsymbol{Y}]_{\leqslant 3}\right), \qquad j \in [\![k+1, n-1]\!] \tag{38}$$

and an associated basis $\mathcal{E}_j$ of $\mathscr{E}_j$. The following proposition is fundamental for the efficiency of the attack.

Proposition 40. *For each $j \in [\![k+1, n-1]\!]$, for each polynomial*

$$\sum_{j'\in[\![k+1,n]\!]\setminus\{j\}} v_{j'}Y_{j'}(X_{j'}+X_j)^2 \in \mathcal{V}_j$$

the polynomial

$$\sum_{j'\in[\![k+1,n]\!]\setminus\{j\}} v_{j'}Y_{j'}(X_{j'}+X_j)$$

cancels on the solution $\boldsymbol{x}, \boldsymbol{y}$. This produces new bilinear polynomials that we can add to our algebraic system.

Proof. Comparing the polynomials in $\mathcal{S}'$ (23) and $\mathcal{B}'$ (36), it is clear that

$$\sum_{j'\in[\![k+1,n]\!]\setminus\{j\}} v_{j'}Y_{j'}(X_{j'}+X_j)^2 \in \mathcal{V}_j \iff \sum_{j'\in[\![k+1,n]\!]\setminus\{j\}} v_{j'}Y_{j'}(Y_{j'}+Y_j)(X_{j'}+X_j)^2 \in \mathcal{E}_j,$$

hence $\dim(\mathcal{E}_j) = m-1$. We can split a polynomial in $\mathcal{E}_j$ in the following way:

$$\begin{aligned} &\sum_{j'\in[\![k+1,n]\!]\setminus\{j\}} v_{j'}Y_{j'}(Y_{j'}+Y_j)(X_{j'}+X_j)^2 \\ &= \sum_{j'\in[\![k+1,n]\!]\setminus\{j\}} v_{j'}Y_{j'}^2(X_{j'}+X_j)^2 + Y_j \sum_{j'\in[\![k+1,n]\!]\setminus\{j\}} v_{j'}Y_{j'}(X_{j'}+X_j)^2. \end{aligned}$$

Since $Y_j\sum_{j'\in[\![k+1,n]\!]\setminus\{j\}} v_{j'}Y_{j'}(X_{j'}+X_j)^2$ is in the ideal generated by $\mathcal{V}_j$, we obtain

$$\sum_{j'\in[\![k+1,n]\!]\setminus\{j\}} v_{j'}Y_{j'}^2(X_{j'}+X_j)^2 \in \mathcal{E}_j + Y_j\mathcal{V}_j.$$

Since the coefficients $v_{j'} \in \mathbb{F}_2$, by applying the Frobenius map $m-1$ times, we get that the polynomial

$$\left(\sum_{j' \in [\![k+1,n]\!] \setminus \{j\}} v_{j'} Y_{j'}^2 (X_{j'} + X_j)^2 = 0 \right)^{2^{m-1}} = \sum_{j' \in [\![k+1,n]\!] \setminus \{j\}} v_{j'} Y_{j'} (X_{j'} + X_j)$$

belong to the ideal generated by $\mathcal{E}_j + \mathcal{V}_j$ and the polynomials $Y_{j'}^{2^m} - Y_{j'}$, $X_{j'}^{2^m} - X_{j'}$ for $j' \in [\![k+1, n]\!]$, hence cancels on the solution. □

As a consequence, from the computation of the $\mathcal{V}_j$'s, and under the assumption that each $\mathcal{V}_j$ contains $m-1$ linearly independent polynomials, we can produce $(m-1)(3m-1)$ bilinear polynomials that are independent, as well as $m-1$ linearly independent linear polynomials in Y_j's (in $\mathcal{V}_n$). A Grobner basis of the system formed by those polynomials together with the polynomials in the systems $\mathcal{V}_j$'s can then be computed, and experimentally all computations are done by staying at degree 3, which gives a global complexity that is polynomial in m. The last steps of the attack (computation of the solutions using the FGLM algorithm, and recovering of the entier support and multiplier) are identical to the $q > 2$ case.

4.8. **Interlacing the algebraic recovering with the filtration.** We now get back to distinguish between the full-length vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ and their shortening due to the filtration attack. We can restore the information lost from the filtration shortening, by simply repeating the attack twice on different sets. This is possible because $2(r-3)$ is very small compared to n. Indeed, if we shorten the positions corresponding to $\mathcal{I}_1 \stackrel{\text{def}}{=} [\![1, r-3]\!]$ (the order is irrelevant) during the filtration attack, at the end of the algebraic recovering we have access to m pairs of vectors

$$\bar{\boldsymbol{x}}_{\mathcal{I}_1} \quad \text{and} \quad \bar{\boldsymbol{y}}_{\mathcal{I}_1} \left(\prod_{i \in \mathcal{I}_1} (\bar{\boldsymbol{x}}_{\mathcal{I}_1} - x_i) \right).$$

Analogously, if we shorten the positions corresponding to $\mathcal{I}_2 \stackrel{\text{def}}{=} [\![(r-3)+1, 2(r-3)]\!]$ during the filtration attack, at the end of the algebraic recovering we have access to m pairs of vectors

$$\bar{\boldsymbol{x}}_{\mathcal{I}_2} \quad \text{and} \quad \bar{\boldsymbol{y}}_{\mathcal{I}_2} \left(\prod_{i \in \mathcal{I}_2} (\bar{\boldsymbol{x}}_{\mathcal{I}_2} - x_i) \right).$$

In particular, if the same specialization has been chosen, we can couple m pairs $(\bar{\boldsymbol{x}}_{\mathcal{I}_1}, \bar{\boldsymbol{x}}_{\mathcal{I}_2})$ such that the two vectors of each pair coincide on the last $n-(r-3)$ coordinates. We can easily detect them from the last $3m$ coordinates, so that we do not need to solve $2m$ linear systems but it is sufficient to choose one pair and solve only the 2 corresponding linear systems. In this way we obtain a full solution $\bar{\boldsymbol{x}}$ for the original problem as

$$\bar{\boldsymbol{x}} = (\ \underbrace{\bar{x}_1, \ldots, \bar{x}_{r-3}}_{\substack{\text{first } r-3 \\ \text{coordinates of } \bar{\boldsymbol{x}}_{\mathcal{I}_1}}}\ , \underbrace{\bar{x}_{r-2}, \ldots, \bar{x}_{2(r-3)}}_{\substack{\text{first } r-3 \\ \text{coordinates of } \bar{\boldsymbol{x}}_{\mathcal{I}_2}}}, \underbrace{\bar{x}_{2(r-3)+1}, \ldots, \bar{x}_{n-3}, 0, 1, \infty}_{\substack{\text{last common coordinates of} \\ \bar{\boldsymbol{x}}_{\mathcal{I}_1} \text{ and } \bar{\boldsymbol{x}}_{\mathcal{I}_2}}}).$$

By replacing the found values in the corresponding $\bar{\boldsymbol{y}}_{\mathcal{I}_1} \left(\prod_{i \in \mathcal{I}_1} (\bar{\boldsymbol{x}}_{\mathcal{I}_1} - x_i) \right)$ and $\bar{\boldsymbol{y}}_{\mathcal{I}_2} \left(\prod_{i \in \mathcal{I}_2} (\bar{\boldsymbol{x}}_{\mathcal{I}_2} - x_i) \right)$, we retrieve $\bar{\boldsymbol{y}}_{\mathcal{I}_1}$ and $\bar{\boldsymbol{y}}_{\mathcal{I}_2}$. Similarly to what done for the

support, we can put together the information of these two vectors and get

$$\bar{\boldsymbol{y}} = (\ \underbrace{\bar{y}_1, \ldots, \bar{y}_{r-3}}_{\substack{\text{first } r-3 \\ \text{coordinates of } \bar{\boldsymbol{y}}_{\mathcal{I}_1}}}\ , \underbrace{\bar{y}_{r-2}, \ldots, \bar{y}_{2(r-3)}}_{\substack{\text{first } r-3 \\ \text{coordinates of } \bar{\boldsymbol{y}}_{\mathcal{I}_2}}},\ \underbrace{\bar{y}_{2(r-3)+1}, \ldots, \bar{y}_{n-1}, 1}_{\substack{\text{last common coordinates of} \\ \bar{\boldsymbol{y}}_{\mathcal{I}_1} \text{ and } \bar{\boldsymbol{y}}_{\mathcal{I}_2}}}\ ).$$

So, a pair of valid support and multiplier has been recovered. However, $\bar{\boldsymbol{x}} \notin \mathbb{F}_{q^m}^n$, because $\bar{x}_n = \infty$. The last question is therefore how to get a valid pair of support and multiplier such that both are defined over $\mathbb{F}_{q^m}$, *i.e.* how to get the alternant representation. In other words, we need to determine some $f \in GL_2(\mathbb{F}_{q^m})$ and $\lambda \in \mathbb{F}_{q^m} \backslash \{0\}$ such that

$$\bar{x}'_i = f(\bar{x}_i) \in \mathbb{F}_{q^m}, \quad \forall i \in [\![1, n]\!]$$

and

$$\bar{y}'_i = \lambda \theta(f, \bar{x}_i)^{r-1} \bar{y}_i \in \mathbb{F}_{q^m}, \quad \forall i \in [\![1, n]\!].$$

We observe that, since there are only $n-1$ coordinates of $\bar{\boldsymbol{x}}$ in $\mathbb{F}_{q^m}$ and $n-1 < q^m$, there exists at least one element $\hat{x} \in \mathbb{F}_{q^m}$ that is different from all $\bar{\boldsymbol{x}}$ coordinates. We also remark that $\hat{x} \neq 0$, since $\bar{x}_{n-2} = 0$, so the map f on $\bar{\mathbb{F}}_{q^m}$

$$f \stackrel{\text{def}}{=} \frac{z}{z - \hat{x}}$$

is induced by an element of the linear group. We have $\theta(f, z) = z - \hat{x}$ if $z \in \mathbb{F}_{q^m}$ and $\theta(f, \infty) = 1$ and we choose $\lambda = 1$. Therefore

$$\begin{aligned} \bar{x}'_i &= \frac{\bar{x}_i}{\bar{x}_i - \hat{x}}, \quad i \in [\![1, n-1]\!], \\ \bar{x}'_n &= 1, \\ \bar{y}'_i &= (\bar{x}_i - \hat{x})^{r-1} \bar{y}_i, \quad i \in [\![1, n-1]\!], \\ \bar{y}'_n &= \bar{y}_n = 1. \end{aligned}$$

We finally obtained a support and a multiplier with coordinates over $\mathbb{F}_{q^m}$ that define the public code. This concludes the key-recovery attack on high-rate random alternant codes.

5. Complexity of the attack

This section analyses the cost of our attack in terms of the parameters n, q, r, m. We estimate the time complexity for both the filtration and the algebraic cryptanalysis separately. The complexity refers to the number of standard operations on the ground field $\mathbb{F}_q$.

5.1. **Complexity of the filtration.**

The core of the first part of the attack consists in the computation of the conductor from Conjecture 18. With the same conditions, and using Proposition **??**, we need to compute a basis for the linear code

$$\left(\mathscr{C} \star \mathscr{D}^{\perp}\right)^{\perp},$$

where $\mathscr{C} \stackrel{\text{def}}{=} \mathscr{A}_r(\boldsymbol{x}_{\mathcal{I}}, \boldsymbol{y}_{\mathcal{I}})^{\perp}$, $\mathscr{D} \stackrel{\text{def}}{=} \mathbf{Sh}_{\mathcal{I}}\left(\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})^{\perp}\right)^{\star 2}$ for $|\mathcal{I}| = 1$, starting from a generator matrix of $\mathscr{A}_r(\boldsymbol{x}, \boldsymbol{y})^{\perp}$. We can choose $\mathcal{I} = \{1\}$ and assume that the matrix is in systematic form. In other words we analyses one iteration of the filtration, assuming that the current alternant code length and degree are n and r respectively.

An upper bound for the total cost can be roughly obtained by multiplying the cost of the first iteration by the number of iterations.

Computing $\mathscr{C}$ can be done in $\mathcal{O}(1)$, since it is enough to drop the i-th position from $\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp$. Moreover $\mathbf{Sh}_1\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)$ can be computed in $\mathcal{O}(1)$ too, whenever the shortened position belongs to the chosen information set, by removing the first row and the first column from the basis, because the first position belongs to the information set. Then the corresponding matrix is still in systematic form. Producing a basis for a component-wise product of two codes $\mathscr{A}$ and $\mathscr{B}$ needs the computation of all possible pairs of basis element in $\mathscr{A}$ and one in $\mathscr{B}$. Therefore it has a linear cost in the length, as well as a linear cost in the dimensions of the two codes $\mathscr{A}$ and $\mathscr{B}$. In the case of a square code, the number of pairs to consider is roughly halved. As a minor improvement, if the code is in systematic form, we can avoid to compute the products in the positions corresponding to the information set for any pair of elements in the basis, since we know them in advance. In particular, since $\dim(\mathbf{Sh}_1\left(\mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})^\perp\right)) = rm-1$, the cost for calculating $\mathscr{D}$ is given by

$$\binom{rm}{2}\cdot(n-rm) = \mathcal{O}\left(r^2m^2n\right).$$

The basis obtained in this way, however, is not in systematic form. Since we want to compute the dual code of $\mathscr{D}$, we need to row reduce its generator matrix. For simplicity, we can bound the dimension of $\mathscr{D}$ with $\binom{rm+1}{2}$ and consequently the complexity of the row reduction step for any value of D with

$$\binom{rm+1}{2}^2 n = \mathcal{O}\left(r^4m^4n\right).$$

Computing $\mathscr{D}^\perp$ then has an additional cost of

$$\left(\binom{rm+1}{2} - D\right)n = \mathcal{O}\left(r^2m^2n\right).$$

Now, since $\dim(\mathscr{D}^\perp) = n - \binom{rm+1}{2} - D$, the product $\mathscr{C}\star\mathscr{D}^\perp$ can be performed in

$$rm\left(n - \binom{rm+1}{2} + D\right)\cdot n = \mathcal{O}\left(rmn^2\right).$$

According to Conjecture 18, the component-wise product of codes obtained is an alternant code of degree $(r-1)m$ and length $n-1$, hence of dimension $n-(r-1)m = \mathcal{O}(n)$. Therefore a row reduced echelon form of its generator matrix can be provided on average in

$$\mathcal{O}\left(n^\omega\right),$$

where $2 \leqslant \omega < 3$ is the constant of linear algebra. In this way the conductor can be computed as the dual of the obtained product with an additional complexity of $\mathcal{O}(rmn)$.

Therefore the cost for computing the conductor is dominated by the computation of $\mathscr{C}\star\mathscr{D}^\perp$ and/or the row reductions. Bearing in mind that we have $r^2m^2 = \mathcal{O}(n)$ in the distinguishable regime, the overall complexity for one iteration of the filtration is given by

$$\mathcal{O}\left(r^2m^2n + n^\omega\right).$$

If we add the costs for each iteration from alternant code degree r down to $q+1$ and taking into account that the length decreases by only 1 at each step, we can

upper bound the total complexity with

$$\sum_{i=q+1}^{r} \left((r-i)mn^2 + n^\omega\right) = \mathcal{O}\left((r-q)(rmn^2 + n^\omega)\right).$$

5.2. **Complexity of algebraic cryptanalysis.** We have seen in Section 4 that the complexity the overall complexity of the algebraic cryptanalysis for an alternant of length n', degree 3 and extension degree m is given by

$$\mathcal{O}\left(m^{2\omega} + n'm\right), \quad n', m \to \infty.$$

We apply this attack on the last alternant code in the filtration, that has length $n' = n - r + 3$ where n is the initial length of the alternant code, and r its degree. As $n' \leqslant n$, the final complexity is

$$\mathcal{O}\left(m^{2\omega} + nm\right), \quad n, m \to \infty.$$

and is smaller than the one of the filtration.

We remark that in order to interlace the filtration with the algebraic resolution, we need to repeat the full attack only twice. Thus, this does not change the order of the complexity. Finally we move from the private key corresponding to the subfield subcode of a Cauchy code to the one of an alternant code. This task has negligible cost with respect to the rest of the algorithm, requiring only $\mathcal{O}(n)$ operations.

Appendix A. Further results about trace codes

We will state and prove here a simple result on the extension of trace codes which will be useful in our context.

Proposition 41. *Let $\mathscr{C} \subseteq \mathbb{F}_{q^m}^n$ be an $\mathbb{F}_{q^m}$- linear code. Then*

$$Tr(\mathscr{C})_{\mathbb{F}_{q^m}} = \sum_{i=0}^{m-1} \mathscr{C}^{q^i}.$$

Proof. Take any $\boldsymbol{c} \in \mathscr{C}$. Then $\mathrm{Tr}(\boldsymbol{c}) = \boldsymbol{c} + \boldsymbol{c}^q + \cdots + \boldsymbol{c}^{q^{m-1}}$ also belongs to $\mathscr{C} + \mathscr{C}^q + \cdots + \mathscr{C}^{q^{m-1}}$. This proves that $\mathrm{Tr}(\mathscr{C}) \subseteq \sum_{i=0}^{m-1} \mathscr{C}^{q^i}$ and therefore $\mathrm{Tr}(\mathscr{C})_{\mathbb{F}_{q^m}} \subseteq \sum_{i=0}^{m-1} \mathscr{C}^{q^i}$. On the other hand, let us prove that any $\mathscr{C}^{q^i}$ is a subspace of $\mathrm{Tr}(\mathscr{C})_{\mathbb{F}_{q^m}}$ for any i. Consider an arbitrary $\mathbb{F}_q$-basis $\alpha \stackrel{\text{def}}{=} \{\alpha_1, \cdots, \alpha_m\}$ of $\mathbb{F}_{q^m}$. Let $\boldsymbol{x}_i \stackrel{\text{def}}{=} \mathrm{Tr}(\alpha_i \boldsymbol{c})$. Since

$$\boldsymbol{x}_i = \alpha_i \boldsymbol{c} + \alpha_i^q \boldsymbol{c}^q + \cdots + \alpha_i^{q^{m-1}} \boldsymbol{c}^{q^{m-1}}$$

we have that

$$\begin{pmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_m \end{pmatrix} = \begin{pmatrix} \boldsymbol{c} & \boldsymbol{c}^q & \cdots & \boldsymbol{c}^{q^{m-1}} \end{pmatrix} \underbrace{\begin{pmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_m \\ \alpha_1^q & \alpha_2^q & \cdots & \alpha_m^q \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_1^{q^{m-1}} & \alpha_2^{q^{m-1}} & \cdots & \alpha_m^{q^{m-1}} \end{pmatrix}}_{\stackrel{\text{def}}{=} \boldsymbol{M}(\alpha)}$$

$\boldsymbol{M}(\alpha)$ is the Moore matrix associated to $\{\alpha_1, \cdots, \alpha_m\}$ and is invertible because the α_i's are linearly independent over $\mathbb{F}_q$. Therefore

$$\begin{pmatrix} \boldsymbol{c} & \boldsymbol{c}^q & \cdots & \boldsymbol{c}^{q^{m-1}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_m \end{pmatrix} \boldsymbol{M}(\alpha)^{-1}$$

and therefore all the $\boldsymbol{c}^{q^i}$ are $\mathbb{F}_{q^m}$-linear combinations of the $\boldsymbol{x}_j$'s and belong therefore to $\mathrm{Tr}\left(\mathscr{C}\right)_{\mathbb{F}_{q^m}}$. This shows that $\mathscr{C}^{q^i} \subseteq \mathrm{Tr}\left(\mathscr{C}\right)_{\mathbb{F}_{q^m}}$ for any i and shows therefore the reverse inclusion

$$\mathscr{C} + \mathscr{C}^q + \cdots + \mathscr{C}^{q^{m-1}} \subseteq \mathrm{Tr}\left(\mathscr{C}\right)_{\mathbb{F}_{q^m}} .$$

□

Appendix B. Proofs of Subsection 3.2

Let us recall the proposition we are going to prove.

Proposition 24. *Let $\mathscr{G}(\boldsymbol{x},\Gamma) \stackrel{def}{=} \mathscr{A}_r(\boldsymbol{x},\boldsymbol{y})$ be a Goppa code of degree r. We have for any code positions i and j with $i \neq j$:*

$$\mathscr{A}_{r+1}(\boldsymbol{x},\boldsymbol{y})^\perp = \mathscr{G}(\boldsymbol{x},\Gamma) + \langle \mathbf{1} \rangle_{\mathbb{F}_q} \quad \textit{(from [Ber00, prop. 1])} \tag{16}$$

$$\mathscr{A}_r(\boldsymbol{x}_i^{\smile}, \boldsymbol{y}_i^{\smile}(\boldsymbol{x}_i^{\smile} - x_i))^\perp = \mathbf{Sh}_i\left(\mathscr{A}_{r+1}(\boldsymbol{x},\boldsymbol{y})^\perp\right) \tag{17}$$

$$\mathbf{Sh}_j\left(\mathscr{A}_r(\boldsymbol{x}_i^{\smile}, \boldsymbol{y}_i^{\smile}(\boldsymbol{x}_i^{\smile} - x_i))^\perp\right) = \mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x}_j^{\smile}, \boldsymbol{y}_j^{\smile}(\boldsymbol{x}_j^{\smile} - x_j))^\perp\right). \tag{18}$$

Proof. **Proof of (16).** (16) was proved in [Ber00, Prop. 1].

Proof of (17).

Choose an $\mathbb{F}_q$-basis $\{\alpha_1, \cdots, \alpha_m\}$ of $\mathbb{F}_{q^m}$. We are first going to prove that

$$\mathscr{G}(\boldsymbol{x}_i^{\smile},\Gamma)^\perp + \langle \mathbf{1} \rangle_{\mathbb{F}_q} = \mathscr{A}_r(\boldsymbol{x}_i^{\smile}, i_{\boldsymbol{y}}^{\smile}(\boldsymbol{x}_i^{\smile} - x_i))^\perp + \langle \mathbf{1} \rangle_{\mathbb{F}_q} . \tag{39}$$

$$\begin{aligned}
\mathscr{A}_r(\boldsymbol{x}_i^{\smile}, \boldsymbol{y}_i^{\smile}(\boldsymbol{x}_i^{\smile} - x_i))^\perp &= \left\langle \mathrm{Tr}\left(\alpha_j \boldsymbol{x}_i^a (\boldsymbol{x}_i^{\smile} - x_i)\boldsymbol{y}_i^{\smile}\right) \mid a \in [\![0, r-1]\!], j \in [\![0, m-1]\!] \right\rangle_{\mathbb{F}_q} \\
&= \left\langle \mathrm{Tr}\left(\alpha_j \boldsymbol{x}_i^{\smile\, a+1} \boldsymbol{y}_i^{\smile}\right) - \mathrm{Tr}\left(\alpha_j x_i \boldsymbol{x}_i^{\smile\, a} \boldsymbol{y}_i^{\smile}\right) \mid a \in [\![0, r-1]\!], j \in [\![0, m-1]\!] \right\rangle_{\mathbb{F}_q} \\
&\subseteq \left\langle \mathrm{Tr}\left(\alpha_j \boldsymbol{x}_i^{\smile\, b} \boldsymbol{y}_i^{\smile}\right) \mid b \in [\![0, r]\!], j \in [\![0, m-1]\!] \right\rangle_{\mathbb{F}_q} \\
&= \mathscr{A}_{r+1}(\boldsymbol{x}_i^{\smile}, \boldsymbol{y}_i^{\smile})^\perp \\
&= \mathscr{G}(\boldsymbol{x}_i^{\smile},\Gamma)^\perp + \langle \mathbf{1} \rangle_{\mathbb{F}_q}
\end{aligned}$$

On the other hand, since $\Gamma(x_i) \neq 0$, then $\Gamma(\boldsymbol{x}_i^{\smile}) \notin \left\langle \boldsymbol{x}_i^{\smile\, a}(\boldsymbol{x}_i^{\smile} - x_i) \mid a \in [\![0, r-1]\!] \right\rangle_{\mathbb{F}_q}$.

Therefore $\left\langle \Gamma(\boldsymbol{x}_i^{\smile}), \boldsymbol{x}_i^{\smile\, a}(\boldsymbol{x}_i^{\smile} - x_i) \mid a \in [\![0, r-1]\!] \right\rangle_{\mathbb{F}_q}$ is a vector space of dimension $r+1$ of evaluations of polynomials with degree at most r. Hence

$$\left\langle \Gamma(\boldsymbol{x}_i^{\smile}), \boldsymbol{x}_i^{\smile\, a}(\boldsymbol{x}_i^{\smile} - x_i) \mid a \in [\![0, r-1]\!] \right\rangle_{\mathbb{F}_q} = \left\langle \boldsymbol{x}_i^{\smile\, b} \mid b \in [\![0, r]\!] \right\rangle_{\mathbb{F}_q} .$$

Since $\mathrm{Tr}\left(\alpha_j \Gamma(\boldsymbol{x}_i^{\smile})\boldsymbol{y}_i^{\smile}\right) = \mathrm{Tr}\left(\alpha_j \cdot \mathbf{1}\right) \in \langle \mathbf{1} \rangle_{\mathbb{F}_q}$, we get

$$\begin{aligned}
\mathscr{G}(\boldsymbol{x}_i^{\smile},\Gamma)^\perp + \langle \mathbf{1} \rangle_{\mathbb{F}_q} &= \left\langle \mathrm{Tr}\left(\alpha_j \boldsymbol{x}_i^{\smile\, b} \boldsymbol{y}_i^{\smile}\right) \mid b \in [\![0, r]\!], j \in [\![0, m-1]\!] \right\rangle_{\mathbb{F}_q} \\
&\subseteq \left\langle \mathrm{Tr}\left(\alpha_j \boldsymbol{x}_i^{\smile\, a} (\boldsymbol{x}_i^{\smile} - x_i)\boldsymbol{y}_i^{\smile}\right) \mid a \in [\![0, r-1]\!], j \in [\![0, m-1]\!] \right\rangle_{\mathbb{F}_q} + \langle \mathbf{1} \rangle_{\mathbb{F}_q} \\
&= \mathscr{A}_r(\boldsymbol{x}_i^{\smile}, \boldsymbol{y}_i^{\smile}(\boldsymbol{x}_i^{\smile} - x_i))^\perp + \langle \mathbf{1} \rangle_{\mathbb{F}_q} .
\end{aligned}$$

Because of the last point, $\mathbf{Sh}_i\left(\mathscr{A}_{r+1}(\boldsymbol{x},\boldsymbol{y})^\perp\right)$ is the set of codewords of $\mathscr{G}(\boldsymbol{x},\Gamma) + \langle \mathbf{1} \rangle_{\mathbb{F}_q}$ which evaluate to 0 at position i. Clearly the elements of $\mathscr{A}_r(\boldsymbol{x}_i^{\smile}, \boldsymbol{y}_i^{\smile}(\boldsymbol{x}_i^{\smile} - x_i))^\perp$ viewed as polynomial evaluations and extended canonically at position i as the linear space

$$\{\mathrm{Tr}\left(\boldsymbol{y}(\boldsymbol{x} - x_i)P(\boldsymbol{x})\right) : \ P \in \mathbb{F}_{q^m}[X], \ \deg P < r\}$$

belong to this set. Since the all one vector does not meet this property and because of (39) this implies the point (17).

Proof of (18)**.** This is just a consequence that $\mathbf{Sh}_i\left(\mathbf{Sh}_j\left(\mathscr{C}\right)\right) = \mathbf{Sh}_j\left(\mathbf{Sh}_i\left(\mathscr{C}\right)\right)$ which holds for any code $\mathscr{C}$. Here we apply it to $\mathscr{C} = \mathscr{A}_{r+1}(\boldsymbol{x},\boldsymbol{y})^\perp$ and apply the previous point:

$$\begin{aligned}\mathbf{Sh}_i\left(\mathscr{A}_r(\boldsymbol{x}_{\check{j}},\boldsymbol{y}_{\check{j}}(\boldsymbol{x}_{\check{j}}-x_j))^\perp\right) &= \mathbf{Sh}_i\left(\mathbf{Sh}_j\left(\mathscr{A}_{r+1}(\boldsymbol{x},\boldsymbol{y})^\perp\right)\right) \text{ (by (17))}\\ &= \mathbf{Sh}_j\left(\mathbf{Sh}_i\left(\mathscr{A}_{r+1}(\boldsymbol{x},\boldsymbol{y})^\perp\right)\right) \text{ (by the previous remark)}\\ &= \mathbf{Sh}_j\left(\mathscr{A}_r(\boldsymbol{x}_{\check{i}},\boldsymbol{y}_{\check{i}}(\boldsymbol{x}_{\check{i}}-x_i))^\perp\right) \text{(by (17))}\end{aligned}$$

□

References

[ABC+20] Martin Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Mizoczki, Ruben Niederhagen, Edoardo Persichetti, Kenneth Paterson, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson, and Wang Wen. Classic McEliece (merger of Classic McEliece and NTS-KEM). `https://classic.mceliece.org`, October 2020. Third round finalist of the NIST post-quantum cryptography call.

[BBB+17] Gustavo Banegas, Paulo S.L.M Barreto, Brice Odilon Boidje, Pierre-Louis Cayrel, Gilbert Ndollane Dione, Kris Gaj, Cheikh Thiécoumba Gueye, Richard Haeussler, Jean Belo Klamti, Ousmane N'diaye, Duc Tri Nguyen, Edoardo Persichetti, and Jefferson E. Ricardini. DAGS : Key encapsulation for dyadic GS codes. `https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/submissions/DAGS.zip`, November 2017. First round submission to the NIST post-quantum cryptography call.

[BCGO09] Thierry P. Berger, Pierre-Louis Cayrel, Philippe Gaborit, and Ayoub Otmani. Reducing key length of the McEliece cryptosystem. In Bart Preneel, editor, *Progress in Cryptology - AFRICACRYPT 2009*, volume 5580 of *LNCS*, pages 77–97, Gammarth, Tunisia, June 21-25 2009.

[Ber00] Thierry P. Berger. On the cyclicity of Goppa codes, parity-check subcodes of Goppa codes and extended Goppa codes. *Finite Fields Appl.*, 6(3):255–281, 2000.

[BJMM12] Anja Becker, Antoine Joux, Alexander May, and Alexander Meurer. Decoding random binary linear codes in $2^{n/20}$: How $1 + 1 = 0$ improves information set decoding. In *Advances in Cryptology - EUROCRYPT 2012*, LNCS. Springer, 2012.

[BLM11] Paulo Barreto, Richard Lindner, and Rafael Misoczki. Monoidic codes in cryptography. In *Post-Quantum Cryptography 2011*, volume 7071 of *LNCS*, pages 179–199. Springer, 2011.

[BLP10] Daniel J. Bernstein, Tanja Lange, and Christiane Peters. Wild McEliece. In Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors, *Selected Areas in Cryptography*, volume 6544 of *LNCS*, pages 143–158, 2010.

[BLP11] Daniel J. Bernstein, Tanja Lange, and Christiane Peters. Wild McEliece Incognito. In Bo-Yin Yang, editor, *Post-Quantum Cryptography 2011*, volume 7071 of *LNCS*, pages 244–254. Springer Berlin Heidelberg, 2011.

[BM17] Leif Both and Alexander May. Optimizing BJMM with Nearest Neighbors: Full Decoding in $2^{2/21n}$ and McEliece Security. In *WCC Workshop on Coding and Cryptography*, September 2017.

[CBB+17] Alain Couvreur, Magali Bardet, Élise Barelli, Olivier Blazy, Rodolfo Canto Torres, Phillipe Gaborit, Ayoub Otmani, Nicolas Sendrier, and Jean-Pierre Tillich. BIG QUAKE. `https://bigquake.inria.fr`, November 2017. NIST Round 1 submission for Post-Quantum Cryptography.

[CC98] Anne Canteaut and Florent Chabaud. A new algorithm for finding minimum-weight words in a linear code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Trans. Inform. Theory*, 44(1):367–378, 1998.

[CCMZ15] Igniacio Cascudo, Ronald Cramer, Diego Mirandola, and Gilles Zémor. Squares of random linear codes. *IEEE Trans. Inform. Theory*, 61(3):1159–1173, 3 2015.

[CFS01] Nicolas Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a McEliece-based digital signature scheme. In *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 157–174, Gold Coast, Australia, 2001. Springer.

[CGG+14] Alain Couvreur, Philippe Gaborit, Valérie Gauthier-Umaña, Ayoub Otmani, and Jean-Pierre Tillich. Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes. *Des. Codes Cryptogr.*, 73(2):641–666, 2014.

[CMCP17] Alain Couvreur, Irene Márquez-Corbella, and Ruud Pellikaan. Cryptanalysis of McEliece cryptosystem based on algebraic geometry codes and their subcodes. *IEEE Trans. Inform. Theory*, 63(8):5404–5418, 8 2017.

[COT14] Alain Couvreur, Ayoub Otmani, and Jean-Pierre Tillich. New identities relating wild Goppa codes. *Finite Fields Appl.*, 29:178–197, 2014.

[Del75] Philippe Delsarte. On subfield subcodes of modified Reed-Solomon codes. *IEEE Trans. Inform. Theory*, 21(5):575–576, 1975.

[Dür87] Arne Dür. The automorphism groups of Reed-Solomon codes. *Journal of Combinatorial Theory, Series A*, 44:69–82, 1987.

[FGLM93] Jean-Charles Faugère, Patrizia M. Gianni, Daniel Lazard, and Teo Mora. Efficient computation of zero-dimensional gröbner bases by change of ordering. *J. Symbolic Comput.*, 16(4):329–344, 1993.

[FGO+11] Jean-Charles Faugère, Valérie Gauthier, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. A distinguisher for high rate McEliece cryptosystems. In *Proc. IEEE Inf. Theory Workshop- ITW 2011*, pages 282–286, Paraty, Brasil, October 2011.

[FGO+13] Jean-Charles Faugère, Valérie Gauthier, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. A distinguisher for high rate McEliece cryptosystems. *IEEE Trans. Inform. Theory*, 59(10):6830–6844, October 2013.

[FOPT10] Jean-Charles Faugère, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. Algebraic cryptanalysis of McEliece variants with compact keys. In *Advances in Cryptology - EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 279–298, 2010.

[GUL09] Valérie Gauthier-Umaña and Gregor Leander. Practical key recovery attacks on two McEliece variants, 2009. IACR Cryptology ePrint Archive, Report2009/509.

[HP03] W. Cary Huffman and Vera Pless. *Fundamentals of error-correcting codes*. Cambridge University Press, Cambridge, 2003.

[KT17] Ghazal Kachigar and Jean-Pierre Tillich. Quantum information set decoding algorithms. In *Post-Quantum Cryptography 2017*, volume 10346 of *LNCS*, pages 69–89, Utrecht, The Netherlands, June 2017. Springer.

[LS01] Pierre Loidreau and Nicolas Sendrier. Weak keys in the McEliece public-key cryptosystem. *IEEE Trans. Inform. Theory*, 47(3):1207–1211, 2001.

[MB09] Rafael Misoczki and Paulo Barreto. Compact McEliece keys from Goppa codes. In *Selected Areas in Cryptography*, Calgary, Canada, August 13-14 2009.

[McE78] Robert J. McEliece. *A Public-Key System Based on Algebraic Coding Theory*, pages 114–116. Jet Propulsion Lab, 1978. DSN Progress Report 44.

[MMT11] Alexander May, Alexander Meurer, and Enrico Thomae. Decoding random linear codes in $O(2^{0.054n})$. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 107–124. Springer, 2011.

[MO15] Alexander May and Ilya Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In E. Oswald and M. Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015*, volume 9056 of *LNCS*, pages 203–228. Springer, 2015.

[MP12] Irene Márquez-Corbella and Ruud Pellikaan. Error-correcting pairs for a public-key cryptosystem. CBC 2012, Code-based Cryptography Workshop, 2012. Available on `http://www.win.tue.nl/~ruudp/paper/59.pdf`.

[MS86] Florence J. MacWilliams and Neil J. A. Sloane. *The Theory of Error-Correcting Codes*. North–Holland, Amsterdam, fifth edition, 1986.

[MT21] Rocco Mora and Jean-Pierre Tillich. On the dimension and structure of the square of the dual of a Goppa code. preprint, 2021.

[Nie86] Harald Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.

[Ran15] Hugues Randriambololona. On products and powers of linear codes under componentwise multiplication. In *Algorithmic arithmetic, geometry, and coding theory*, volume 637 of *Contemp. Math.*, pages 3–78. Amer. Math. Soc., Providence, RI, 2015.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[Sen00] Nicolas Sendrier. Finding the permutation between equivalent linear codes: The support splitting algorithm. *IEEE Trans. Inform. Theory*, 46(4):1193–1203, 2000.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In S. Goldwasser, editor, *FOCS*, pages 124–134, 1994.

[SS92] Vladimir Michilovich Sidelnikov and S.O. Shestakov. On the insecurity of cryptosystems based on generalized Reed-Solomon codes. *Discrete Math. Appl.*, 1(4):439–444, 1992.

[Ste88] Jacques Stern. A method for finding codewords of small weight. In G. D. Cohen and J. Wolfmann, editors, *Coding Theory and Applications*, volume 388 of *LNCS*, pages 106–113. Springer, 1988.

LITIS, University of Rouen Normandie
Email address: magali.bardet@univ-rouen.fr

Inria, 2 rue Simone Iff, 75012 Paris, France, Sorbonne Universités, UPMC Univ Paris 06
Email address: rocco.mora@inria.fr

Inria, 2 rue Simone Iff, 75012 Paris, France
Email address: jean-pierre.tillich@inria.fr