Syndrome decoding for Hermite codes with a Groebner-basis algorithm *

Irene I. Bouw, Sabine Kampf

Abstract

This paper gives a new approach to decoding Hermite codes using the key equation, avoiding the use of majority voting. Our approach corrects up to $(d_{\min} - 1)/2$ errors, and works up to some extent also beyond. We present an efficient implementation of our algorithm based on a Groebner-basis-type iterative procedure.

Keywords: Hermite codes, algebraic decoding, Groebner bases.
Mathematics Subject Classification (2000): Primary 14G50; Secondary 11T71.

Introduction

Algebraic-geometry codes (AG-codes) are a generalization of Reed–Solomon codes (RS-codes), and have many nice properties ([8], [24], [25]). For example, the family of AG-codes includes a rich variety of codes which are better than the Varshamov–Gilbert bound ([26]). Several approaches to decoding of AG-codes can be found in the literature (for example [12], [21], [23], [9], [1], [18]). However, up to now AG-codes are not used in practical applications. One of the reasons appears to be that understanding the proposed algorithms requires quite some background in algebraic geometry.

The starting point of this paper is the well-known key equation for decoding of AG-codes. In contrast to the situation for Reed–Solomon codes, there exist many forms of the key equation for AG-codes (for example [19], [22], [6], [5], [11], [1], [18]. See [18, Section 3.5] for an overview of all approaches). These decoding algorithms correct up to $(d_{\min} - 1)/2 - s$ errors where s is the Clifford defect (Section 3) and $d_{\min}$ the minimum distance. There exists an extension which corrects up to $(d_{\min} - 1)/2$ errors. It uses *majority voting* to estimate certain unknown syndromes, which are needed in the decoding (see e.g. [7], [10, Chapter 6.3]).

In this paper, we present a new approach to decoding up to $(d_{\min} - 1)/2$ errors, which does not use the unknown syndromes (and hence does not need

*This work was supported by the German Research Council "Deutsche Forschungsgemeinschaft" (DFG) under Grant No. Bo867/22.

majority voting). We define an approximate version of the key equation, which we call the *modified key equation* and we show that it can be used for decoding. For more than $(d_{\min}-1)/2-s$ errors, not all solutions of the key equation may be used for decoding. However, most false solutions are easily recognized.

The second part of the paper discusses algorithmic aspects. We present a Groebner-basis approach to computing solutions of the (modified) key equation. This approach is essentially a simplification of the subresultant method of [22]. In contrast to, for example, [12] and [18] we compute the error-locator polynomial of smallest degree rather than a basis of the error-locator ideal. This significantly decreases the number of iterations of the algorithm. The overall complexity of our algorithm however is, at least for practical purposes, the same as that using majority voting. In the last section, we briefly discuss an extension of the algorithm for correcting more than $(d_{\min}-1)/2$ errors (Section 7). Using a concrete implementation, we tested the practicality of the proposed algorithm.

Our presentation of the material is as elementary and self-contained as possible, formulating results from algebraic geometry as a black box. Reading the paper requires only limited knowledge of algebraic geometry (the statement of the Riemann–Roch Theorem and a working knowledge of local rings and valuations). The main results are formulated as easy to use and to implement statements on polynomials. The restriction to Hermite curves also makes the paper more accessible. This seems no great restriction, as Hermite codes are most likely to be among the first AG-codes that become relevant for practical applications. It seems relatively straightforward to adapt the main idea of the algorithm to one-point codes or even general AG-codes (e.g. by comparing our approach to the set-up of [19] and [18]). However, we do not claim to have worked out all details.

Modeling the approach on the well-known case of Reed–Solomon codes, allows the reader to easily recognize the differences and similarities between both cases. Our approach is modeled on the usual Sugiyama algorithm which is based on the Euclidean algorithm for RS-codes. (This in contrast to for example the approach of O' Sullivan–Bras-Amarós, which is a generalization of Kötter's version of the Berlekamp–Massey algorithm ([18, Section 3.3.6]).

We now describe our algorithm in more detail. Let $\mathcal{X}_q$ be the Hermite curve over $\mathbb{F}_{q^2}$ given by the affine equation $x^{q+1}=y^q+y$. It is known that $\mathcal{X}_q(\mathbb{F}_{q^2})$ has cardinality q^3+1. We denote by P the unique point at infinity, and by $\{P_1,\ldots,P_n\}:=\mathcal{X}_q(\mathbb{F}_{q^2})\setminus\{P\}$. For $2g(\mathcal{X}_q)-1\leq m<n=q^3$, we denote by $\mathcal{C}$ the image of

$$\varphi: L(mP)\to\mathbb{F}_p^m,\quad f\mapsto(f(P_i))_i,$$

where $L(mP)$ denotes the Riemann–Roch space as $\mathbb{F}_{q^2}$-vector space. We define the degree of elements of $L(mP)$ by minus the valuation at P (Section 1).

Given a received word $\boldsymbol{r}$, the first step of the decoding algorithm computes a minimal error-locator polynomial $\Lambda\in\cup_\nu L(\nu P)$. Here minimality is considered with respect to the degree. We show that Λ is a solution to a (modified) key equation

$$\tilde{S}\cdot\Lambda\equiv R\pmod{y^{b_m+1}},$$

where the degree of R is bounded in terms of the degree of Λ (see Section 4 for a precise statement). Here $\tilde{S}$ is the syndrome polynomial, which can be explicitly computed from $\boldsymbol{r}$ (Section 4). To explain the relation between the modified and the usual key equation, we present the latter in Section 3 in a way suitable for our purposes.

We give an iterative algorithm for computing Λ and R, which is similar in spirit to the Euclidean algorithm, which is used for this task in the Reed–Solomon case. Since the ring $\mathcal{R}$ of functions we work in here is not Euclidean, we need to use a division algorithm similar to what is used in the Groebner-basis algorithm(Section 5). After the computation of an error-locator polynomial Λ, we find the corresponding error values by computing the residues of R/Λ at the error positions. This step is analogous to the well-known Forney formula for RS-codes ([17, Section 10.2]). For AG-codes a similar formula can be found for example in [16].

Algorithm 4.9 describes two variants of this selection procedure. One variant (d*) is fast and mostly works, the other variant (d) is a bit slower but always works. In Section 5.1, we present simulations based on a MAGMA implementation illustrating the efficiency of variant (d*). Running times are analyzed in Section 6.

In Section 7, decoding beyond half the minimum distance is considered. We show how to obtain a basis for all solutions to the key equation and calculate its size. We also discuss why it is not feasible to try to decode this larger number of errors without further information about the error. In a subsequent paper [15], we present some methods that allow efficient decoding beyond half the minimum distance.

1 Preliminaries

Let $\mathbb{F}_{q^2}$ be a finite field and $\mathcal{X}_q$ the Hermite curve over $\mathbb{F}_{q^2}$, i.e. the projective curve defined by the affine equation

$$x^{q+1} = y^q + y.$$

Recall that the genus of $\mathcal{X}_q$ is $g := q(q-1)/2$. Let P be the unique point of $\mathcal{X}_q$ at infinity. We denote by

$$\mathcal{R} = \cup_{m \geq 0} L(mP)$$

the *affine ring* of $\mathcal{X}_q$ at P. For convenience, we refer to the elements of $\mathcal{R}$ as *polynomials*. It is well-known that

$$\Phi := \{\varphi_{a,b} := x^a y^b \mid 0 \leq a \leq q, 0 \leq b\}$$

is a basis of $\mathcal{R}$.

For $f \in \mathrm{Frac}(\mathcal{R})$, we define $\rho(f) = -\mathrm{ord}_P(f)$. Alternatively, one may define ρ on monomials by $\rho(x^a y^b) = -\mathrm{ord}_P(x^a y^b) = qa + (q+1)b$ and extend ρ to $f = \sum_{a,b} f_{a,b} x^a y^b \in \mathcal{R}$ by defining

$$\rho(f) = \max_{(a,b): f_{a,b} \neq 0} \rho(x^a y^b),$$

and $\rho(f/g) = \rho(f) - \rho(g)$ for $f/g \in \mathrm{Frac}(\mathcal{R})$.

In analogy to the situation for $\mathbb{P}^1$, we refer to $\rho(f)$ as the *degree* of f. The term $f_{a,b}x^a y^b$ with $\rho(x^a y^b) = \rho(f)$ is called the *leading term of* f. If the leading term of f is $x^a y^b$, we call f *monic*.

It is well known that for every $r \in \mathbb{N}$ there is at most one monomial $x^a y^b \in \Phi$ with $\rho(x^a y^b) = r$. Therefore we can order the monomials $\varphi_{a,b}$ according to their ρ-value. We sometimes also write $\varphi_0 = \varphi_{0,0} = 1, \varphi_1 = \varphi_{1,0} = x, \varphi_2 = \varphi_{0,1} = y, \ldots$ to refer to this ordering.

We denote by $\{P_0 = (0,0), \ldots, P_{n-1}\} = \mathcal{X}_q(\mathbb{F}_{q^2}) \setminus \{P\}$, and let $D := \sum_{i=0}^{n-1} P_i$. It is well known that $n = q^3$. For $2g - 1 \leq m < n$, we consider the code $\mathcal{C} := \mathcal{C}_L(D, mP)$, which is defined as the image of

$$L(mP) \to \mathbb{F}^n, \qquad f \mapsto (f(P_i))_{i=0..n-1}.$$

We denote by $k = m - g + 1$ the dimension of the code, by $d_{\min}$ the minimum distance of the code, $d^* = n - m$ the designed minimum distance, and by $k^\perp$ the dimension of the dual code. In our situation, the dual code is isomorphic to $L(m^\perp P)$, where $m^\perp = n + 2g - 2 - m$. In particular, the check matrix of $\mathcal{C}$ is

$$H := \Big(\varphi_i(P_j)\Big)_{0 \leq i \leq k^\perp, 0 \leq j \leq n-1},$$

where also $k^\perp = m^\perp - g + 1$. The minimum distance $d_{\min}$ is computed in [27] (see also [10, Section 5.3]). To avoid a case distinction, we sometimes rather use the designed minimum distance.

2 Syndrome decoding and the key equation

In this and the next section, we review the well-known key equation for Hermite codes, formulating it in a way convenient for our purposes. We mainly follow [19] including some simplifications and corrections from [6] (see also [11]).

Suppose that $\boldsymbol{r} = \boldsymbol{c} + \boldsymbol{e} = (r_i)$ is a received vector, where $\boldsymbol{c} = (c_i)$ is a codeword and $\boldsymbol{e} = (e_i)$ the corresponding error vector. Let $t = \mathrm{wt}(\boldsymbol{e})$ be the number of errors. Denote by $I \subset \{0, 1, \ldots, n-1\}$ the error positions and by $Q = \sum_{i \in I} P_i$ the error divisor. An *error-locator polynomial* Λ is a function $\Lambda \in \mathcal{R}$ such that $\mathrm{ord}_{P_i}\Lambda > 0$ for all $i \in I$. An error-locator polynomial of minimal degree is called a *minimal error locator*. The set of all error-locator polynomials forms an ideal I_e in $\mathcal{R}$ which is called the *error-locator ideal*.

Definition 2.1 For every $\varphi_i \in \Phi$, we define a *syndrome element* as

$$s_i = \sum_{j=0}^{n-1} e_j \varphi_i(P_j) = s_{a,b} \text{ if } \varphi_i = \varphi_{a,b}.$$

For $m^\perp$ as in Section 1, we put $a_m = q$, $b_m = \max\{b \mid x^a y^b \in \Phi, \rho(x^a y^b) \leq m^\perp\}$. We define the *syndrome polynomial* by

$$S := \sum_{0 \leq a \leq a_m, 0 \leq b \leq b_m} s_{a,b} x^{a_m - a} y^{b_m - b} \in \mathcal{R}.$$

We remark that for $\rho(x^a y^b) \leq m^\perp$, we may compute the syndromes $\boldsymbol{s} := (s_{a,b})_{\rho(x^a y^b) \leq m^\perp}$ from the check matrix and the received vector, since $\boldsymbol{H}\boldsymbol{r}^t = \boldsymbol{H}\boldsymbol{e}^t = \boldsymbol{s}$. Sometimes, these syndromes are called the *known syndromes* in contrast to the *unknown syndromes* $s_{a,b}$ for $\rho(x^a y^b) > m^\perp$. In Section 4 we will see that these unknown syndromes are not actually used in our decoding algorithm.

The following lemma gives an interpretation of the syndrome polynomial, which plays a central role in our decoding algorithm. Basically the lemma states that the syndrome polynomial S is an approximation of a rational function U defined below, from which we may easily read off the error locations, and the error values.

For every $0 \leq i < n$ we define a rational function u_i, as follows. We write $P_i = (\alpha_i, \beta_i)$ and define

$$u_i = \frac{y^q + y - \alpha_i^{q+1}}{(x - \alpha_i)(y - \beta_i)} = \frac{1 + \sum_{j=0}^{q-1} (y^j \beta_i^{q-1-j})}{x - \alpha_i}. \tag{1}$$

Note that u_i has a simple pole in P_i, a pole in P with $\rho(u_i) = q^2 - q - 1$, and no other poles. Moreover, the polar part of u_i at P_i is $1/(x - \alpha_i)$. Put

$$U := -\sum_{i \in I} e_i \beta_i^{b_m + 1} u_i. \tag{2}$$

Since $P_0 = (0, 0)$ is the only point of the curve with $y = 0$, the set of poles of U is exactly $\{i \in I \mid i \neq 0\}$. For $i \neq 0$, the polar part of U at P_i is

$$-\frac{e_i \beta_i^{b_m + 1}}{x - \alpha_i}.$$

This observation will be used in Section 4 to calculate the error values. The point $P_0 = (0, 0)$ needs to be treated differently from the other points since all basis functions $\varphi_{a,b}$ for $(a, b) \neq (0, 0)$ vanish at P_0.

Lemma 2.2 *The syndrome polynomial may also be written as*

$$S(x, y) = \sum_{i \in I} e_i (y^{b_m + 1} - \beta_i^{b_m + 1}) u_i.$$

Proof: Using the definition of the syndromes and (1), we can rewrite S as

$$\begin{aligned} S &= \sum_{0 \leq a \leq q, 0 \leq b \leq b_m} \Big(\sum_{i \in I} e_i \alpha_i^{a_m - a} \beta_i^{b_m - b}\Big) x^a y^b = \sum_{i \in I} e_i \Big(\sum_{u=0}^{b_m} \beta_i^{b_m - u} y^u\Big)\Big(\sum_{v=0}^{q} \alpha_i^{q - v} x^v\Big) \\ &= \sum_{i \in I} e_i \frac{(y^{b_m + 1} - \beta_i^{b_m + 1})(y^q + y - \alpha_i^{q+1})}{(x - \alpha_i)(y - \beta_i)} = \sum_{i \in I} e_i (y^{b_m + 1} - \beta_i^{b_m + 1}) u_i. \end{aligned}$$

□

Our definition of S mimics the usual definition of the syndrome polynomial for RS-codes. In the literature one finds many definitions of similar objects, which are sometimes also rational functions or differential forms. For example, our polynomial S is an approximation of a transformation of the rational function in [18, Lemma 3.14] (compare also Lemma 2.2 with [18, (3.30)]).

The proof of the following proposition uses an idea from [12].

Proposition 2.3 *Let $\Lambda \in \mathcal{R}$ be an error-locator polynomial. Then there exists a polynomial $R \in \mathcal{R}$ such that*

$$(3) \qquad \begin{cases} \operatorname{ord}_{P_0}(\Lambda \cdot S - R) & \geq \operatorname{ord}_{P_0}(y^{b_m+1}), \\ \rho(R) - \rho(\Lambda) & \leq qa_m + (q+1)b_m - m^{\perp} - 1 =: \ell. \end{cases}$$

Proof: Let $\Lambda = \sum_{a,b} \lambda_{a,b}\varphi_{a,b}$ be an error-locator polynomial, and let $\mu := \rho(\Lambda)$. We write $\Lambda \cdot S = \sum_{a,b} t_{a,b}\varphi_{a,b}$.

We define

$$R := \sum_{qa+(q+1)b \leq \ell+\mu} t_{a,b}x^a y^b.$$

Then obviously $\rho(R) \leq \ell + \mu$. Consider the coefficient $t_{a,b}$ of $\Lambda \cdot S$ for

$$(4) \qquad \mu + \ell < \rho(x^a y^b) \text{ and } b \leq b_m,$$

i.e.

$$(5) \qquad t_{a,b} = \sum \lambda_{i,j} s_{a_m-a+i, b_m-b+j}.$$

We remark that for all (i,j) with $\lambda_{i,j} \neq 0$ we have that $\rho(x^i y^j) \leq \rho(\Lambda) = \mu$. Together with (4) and the definition of ℓ, this implies that for all such (i,j) we have

$$q(a_m - a + i) + (q+1)(b_m - b + j) < \mu + qa_m + (q+1)b_m - \ell - \mu = m^{\perp} + 1.$$

Therefore for (i,j) in this range, the syndromes s_{a_m-a+i,b_m-b+j} are known syndromes. This implies that we may rewrite (5) as

$$(6) \qquad t_{a,b} = \sum_{u=0}^{n-1} e_u \varphi_{a_m-a,b_m-b}(P_u)\Lambda(P_u).$$

Note that if $u \in \{0, \ldots, n-1\}$ and $e_u \neq 0$, then $\Lambda(P_u)$ vanishes, since Λ is an error-locator polynomial. Equation (6) implies now that $t_{a,b} = 0$ for all (a,b) satisfying (4). Since we defined $a_m = q$, it follows from this observation and the definition of R that the polynomial $T := \Lambda \cdot S - R$ contains only monomials $x^a y^b$ with $b > b_m$. We conclude that $\operatorname{ord}_{P_0}(T) \geq \operatorname{ord}_{P_0}(y^{b_m+1})$. □

Definition 2.4 Let Λ, R be arbitrary elements of $\mathcal{R}$. We say that (Λ, R) is a *solution of the key equation* if both conditions of (3) are satisfied.

One easily shows that in $\mathcal{R}$ the first part of the key equation (3) is equivalent to

$$y^{b_m+1} \mid (\Lambda S - R),$$

or in a notation that is more similar to that used for RS-codes

$$\Lambda \cdot S \equiv R \pmod{y^{b_m+1}}.$$

The following lemma implies that for $\Lambda \in \mathcal{R}$ there exists at most one solution R with $\rho(R) \leq \rho(y^{b_m+1})$ such that (Λ, R) is a solution of the key equation. We formulate it slightly more generally, since this will be used in Section 5.

Lemma 2.5 *Let $\Lambda \in \mathcal{R}$. There exists a unique polynomial $R \in \mathcal{R}$ such that*

$$\begin{aligned} \operatorname{ord}_{P_0}(\Lambda S - R) \geq \operatorname{ord}_{P_0}(y^{b_m+1}), \qquad & \textit{and} \\ \rho(R) \quad & \textit{minimal.} \end{aligned}$$

Proof: We remark that $\operatorname{ord}_{P_0}(\varphi_{a,b}) = a + (q+1)b$. Therefore all $\varphi_{a,b} \in \Phi$ have distinct values $\operatorname{ord}_{P_0}(\varphi_{a,b})$. This immediately implies the existence of a polynomial R as in the statement of the lemma: R is the unique polynomial which is congruent to $\Lambda S \pmod{y^{b_m+1}}$ which does not contain any monomials with $\rho(\varphi_{a,b}) \geq \rho(y^{b_m+1}) = (q+1)(b_m+1)$. The uniqueness of R follows from this description. $\square$

We end this section with an easy lemma on the degree of a minimal error-locator polynomial.

Lemma 2.6 *Let $t = |I|$ be the number of errors and Λ a minimal error-locator polynomial. Let N be minimal such that the dimension $\ell(NP)$ of the Riemann–Roch space satisfies $\ell(NP) > t$. Then*

$$t \leq \rho(\Lambda) \leq N \leq t + g.$$

Proof: The first inequality follows since the number of zeros of Λ is less than or equal to its degree. The second follows by elementary linear algebra. The last follows from the Riemann–Roch Theorem. $\square$

3 Solutions to the key equation

The main result of this section is Corollary 3.2, which is a partial converse to Proposition 2.3. We assume that (Λ, R) is a solution of the key equation. Then Corollary 3.2 states that the solution Λ is an error-locator polynomial, provided the number of errors t is small enough.

Proposition 3.1 *Let (Λ, R) be a solution to the key equation. Write $\mu = \rho(\Lambda)$. Then*

$$R - \Lambda U \in L((\mu + \ell)P + Q - (q+1)(b_m+1)P_0).$$

Proof: The definition of U (Equations (1) and (2)) implies that the poles of $R-\Lambda U$ are contained in $\{P_i\}_{i\in I}\cup\{P\}$. Moreover, in P_i with $i\in I\setminus\{0\}$, the rational function $R-\Lambda U$ has at most a simple pole. The order of the pole in P equals $-\rho(R-\Lambda U)$. Since $\rho(\Lambda)=\mu$ by definition, we conclude that

$$\rho(R-\Lambda U)\leq \rho(\Lambda)+\max\left(\rho\left(\frac{R}{\Lambda}\right),\rho(U)\right).$$

Since (Λ,R) is a solution to the key equation (3), it follows that $\rho(R)-\rho(\Lambda)\leq \ell$. The definition of U implies that $\rho(U)\leq (q-1)(q+1)-q=q^2-q-1=2g-1$. We conclude that

$$\rho(R-\Lambda U)\leq \mu+\max(\ell,2g-1).$$

The definition of b_m implies that

$$(q+1)b_m\leq m^{\perp}\leq (q+1)b_m+q,$$

therefore $2g-1\leq \ell$. We conclude that $\rho(R-\Lambda U)\leq \mu+\ell$.

It remains to estimate $\mathrm{ord}_{P_0}(R-\Lambda U)$. Lemma 2.2 states that

$$S-U=(\sum_{i\in I}e_iu_i)y^{b_m+1}.$$

We conclude that $\mathrm{ord}_{P_0}(S-U)\geq \mathrm{ord}_{P_0}(y^{b_m+1})=(q+1)(b_m+1)$ if $0\notin I$. In the case that $0\in I$, we have $u_0=(y^{q-1}+1)/x$, and hence

$$S-U=e_0y^{b_m}x^q+\sum_{i\in I\setminus\{0\}}e_iu_iy^{b_m+1}. \tag{7}$$

We conclude that $\mathrm{ord}_{P_0}(S-U)=(b_m+1)(q+1)-1$.

Since (Λ,R) is a solution to the key equation (3), it follows that $T=S\Lambda-R$ satisfies $\mathrm{ord}_{P_0}(T)\geq (q+1)(b_m+1)$. We may write $R-\Lambda U=\Lambda(S-U)-T$. Therefore it follows that

$$\mathrm{ord}_{P_0}(R-\Lambda U)\geq (q+1)(b_m+1)-1.$$

□

The next corollary gives a necessary condition for a solution to the key equation to be an error-locator polynomial. This result is also proved in [19, Prop. 14] (see also [4, Section 1.4.6]). Recall that $t=\deg(Q)$ is the number of errors. To state it, we need to first recall the definition of the Clifford defect. For $\nu\in\mathbb{Z}$, we consider the divisor νP and define the *defect* by

$$s_\nu:=\frac{\deg(\nu P)}{2}-(\ell(\nu P)-1).$$

From the Riemann–Roch Theorem it follows that $s_\nu=s_{2g-2-\nu}$, since νP and $K-\nu P$ have the same defect. It follows therefore that for $\nu\leq 0$ or $\nu\geq 2g-2$ we have that $s_\nu \leq 0$. An elementary computation ([19, Prop. 16]) in the remaining case $0 < \nu < 2g-2$ now shows that

$$s_\nu \leq \begin{cases} (q-1)^2/8 + 1/2, & \text{if } q \equiv 1 \pmod 2, \\ (q-2)^2/8 + 1/2, & \text{if } q \equiv 0 \pmod 2. \end{cases}$$

The latter rational number is called the *Clifford defect* $s(\mathcal{X}_q)$ of the Hermite curve $\mathcal{X}_q$.

Corollary 3.2 *Let (Λ, R) be a solution of the key equation (3) with $t \leq (d^* - 1)/2 - s(\mathcal{X}_q)$. Then Λ is an error-locator polynomial.*

Proof: Let (Λ, R) be as in the statement of the corollary, and define U as before. Define $\mu = \rho(\Lambda)$. We first assume that $t + \mu < d^*$. Lemma 2.6 implies that for $t < (d^* - g)/2$ there exists an error-locator polynomial Λ with $\rho(\Lambda) = \mu$ satisfying this inequality. Proposition 2.3 implies that there exists an R such that (Λ, R) is a solution to the key equation (3).

Proposition 3.1 implies that $R - U\Lambda \in L(Q + (\mu+\ell)P - (q+1)(b_m+1)P_0)$, where $\mu = \rho(\Lambda)$. Since

$$\begin{aligned} \deg(Q + (\mu+\ell)P - (q+1)(b_m+1)P_0) =& t + \mu + \ell - (q+1)(b_m+1) \\ <& d^* + q^2 - q - 1 - m^\perp - 1 = 0, \end{aligned}$$

it follows from the Riemann–Roch Theorem that

$$L(Q + (\mu+\ell)P - (q+1)(b_m + 1)P_0) = \{0\}.$$

Now let $\mu = \rho(\Lambda)$ be arbitrary, and define

$$\Delta_\mu = Q + (\mu+\ell)P - (q+1)(b_m + 1)P_0.$$

We claim that for $t < (d^* - 1)/2 - s$ the space $L(\Delta_\mu)$ is still trivial. We prove this statement inductively raising μ by one in each step as long as the condition $t < (d^* - 1)/2 - s$ is still satisfied. We define $A = \mu P - Q$ and $B = \Delta_{\mu+1}$. Note that $A + B \sim \deg(A+B)P = (2\mu + 1 - d^*)P$. It therefore follows that

$$\begin{aligned} \ell(A+B) &= \deg(A+B)/2 - s(A+B) + 1 \\ &\geq \deg(A+B)/2 - s(\mathcal{X}_q) + 1 = \mu - (d^* - 1)/2 - s + 1 > \deg(B). \end{aligned}$$

The last inequality uses the assumption on t. We now apply [4, Lemma 1.52]. This lemma states that for any divisors A, B such that $\deg(B) < \ell(A+B)$ and $\ell(B) \neq 0$, we have that $\ell(A) \neq 0$. In our situation, if Λ is not an error-locator polynomial, it follows that $\ell(A) = 0$, and hence that $\ell(B) = \ell(\Delta_{\mu+1}) = 0$. This proves the claim.

The statement $L(\Delta_\mu) = \{0\}$ implies that

$$\frac{R}{\Lambda} = U. \tag{8}$$

Since U has poles in $I \setminus \{0\}$, it follows immediately that all error locations P_i, except possibly P_0, are zeros of Λ.

It remains to determine whether P_0 is an error location. We assume that $e_0 \neq 0$ and $\Lambda(P_0) \neq 0$. Equation (7) implies that the value of

$$\frac{S-U}{y^{b_m}x^q}$$

at P_0 is e_0 which is nonzero. In particular this implies that $S - U - e_0 y^{b_m} x^q$ has valuation at P_0 greater than or equal to $(b_m+1)(q+1)$. Recall that this means that this term is divisible by y^{b_m+1}. As in the proof of Proposition 3.1, we write $R - \Lambda U = \Lambda(S-U) - T$. Since T is also divisible by y^{b_m+1}, the assumption that $\Lambda(P_0) \neq 0$ implies that $\mathrm{ord}_{P_0}(R-\Lambda U) = \mathrm{ord}_{P_0}(e_0 y^{b_m} x^q) < (q+1)(b_m+1)$. Hence in particular, $R - \Lambda U$ is nonzero. But this contradicts the fact that $R - \Lambda U = 0$ (8). We conclude that $\Lambda(P_0) = 0$. Hence Λ is an error-locator polynomial. □

Note that in the situation of Corollary 3.2 the error values can be easily read off from $R/\Lambda = U$ and the definition of U. The corollary unfortunately only applies when the number of errors is relatively small. In the next section, we discuss an extension of the idea which corrects up to $(d^*-1)/2$ errors. Example 4.7.(b) below illustrates that this is indeed more general.

The following proposition explores the potential of the key equation for correcting errors even beyond half the minimum distance. The bottleneck for the algorithm is assuring whether a solution of the key equation indeed is an error-locator polynomial. We discuss this question in more detail in the next section. The error values are determined in Lemma 4.3.

Proposition 3.3 *Suppose that Λ is an error-locator polynomial with $\rho(\Lambda) < d^*$. Let (Λ, R) be the corresponding solution of the key equation. Then*

$$\frac{R}{\Lambda} = U.$$

In particular, the error locations $i \neq 0$ are exactly the simple poles of R/Λ.

Proof: We suppose that Λ is an error-locator polynomial with $\mu := \rho(\Lambda) < d^*$. Then ΛU does not have poles outside P. Proposition 3.1 implies that

$$R - \Lambda U \in L((\mu+\ell)P - (q+1)(b_m+1)P_0).$$

The assumption on μ implies that

$$\deg((\mu+\ell)P - (q+1)(b_m+1)P_0) = \mu + q^2 - q - 2 - m^\perp = \mu - d^* < 0.$$

As in the proof of Proposition 3.1, we conclude that

$$\frac{R}{\Lambda} = U.$$

□

4 The modified key equation

In this section, we address the problem that the syndrome polynomial S as defined in Section 2 not only involves those syndromes which can be computed in terms of the received vector but also so-called unknown syndromes $s_{a,b}$ with $\rho(x^a y^b) > m^\perp$. The following lemma shows that to compute an error-locator polynomial Λ it suffices to use the known syndromes. To show this, we define the *modified syndrome polynomial* as :

$$\tilde{S} := \sum_{\rho(x^a y^b) \le m^\perp} s_{a,b} x^{a_m - a} y^{b_m - b}.$$

We say that $(\Lambda, \tilde{R}) \in \mathcal{R}^2$ satisfies the *modified key equation* if

$$\begin{cases} \mathrm{ord}_{P_0}(\Lambda \cdot \tilde{S} - \tilde{R}) & \ge \mathrm{ord}_{P_0}(y^{b_m+1}), \\ \rho(\tilde{R}) - \rho(\Lambda) & \le q a_m + (q+1) b_m - m^\perp - 1 =: \ell. \end{cases} \tag{9}$$

Lemma 4.1 *For $R \in \mathcal{R}$ define $\tilde{R} = R - \Lambda(S - \tilde{S})$. Then (Λ, R) is a solution to the key equation (3) if and only if $(\Lambda, \tilde{R})$ is a solution to the modified key equation.*

Proof: This follows immediately from the observation that $S - \tilde{S}$ only contains monomials $x^a y^b$ with $\rho(x^a y^b) < q^2 + (q+1)b_m - m^\perp$. □

Remark 4.2 The above lemma immediately implies that the statement of Lemma 2.5 also holds for the modified key equation.

The following lemma is the modified analog to Corollary 3.2.

Lemma 4.3 *Let $(\Lambda, \tilde{R})$ be a solution to the modified key equation. Assume that $t \le (d^* - 1)/2 - s(\mathcal{X}_q)$.*

(a) Then $i \in I \setminus \{0\}$ if and only if $\tilde{R}/\Lambda$ has a simple pole in P_i. More precisely, the polar part of $\tilde{R}/\Lambda$ in $P_i = (\alpha_i, \beta_i)$ is

$$-\frac{e_i \beta_i^{b_m+1}}{x - \alpha_i}.$$

(b) We have that

$$e_0 = s_{0,0} - \sum_{j=1}^{n-1} e_j.$$

In particular, $i = 0$ is an error position if and only if $s_{0,0} \ne \sum_{j=1}^{n-1} e_j$.

Remark 4.4 As an alternative for the criterion of Lemma 4.3.(b) one may also use that

$$R - \Lambda U \equiv e_o y^{b_m} x^q \pmod{y^{b_m+1}}$$

(compare to the proof of Corollary 3.2.)

Proof: We note that

$$\frac{\tilde{R}}{\Lambda} = \frac{R}{\Lambda} + \tilde{S} - S,$$

where $\tilde{S} - S$ is a polynomial. Statement (a) follows immediately from this observation together with the characterization of the error locations different from P_0 as poles of R/Λ. Statement (b) follows immediately from the definition of the syndrome $s_{0,0}$. □

Summarizing, we have shown that error-locator polynomials always correspond to solutions of the modified key equation. Under the assumption $t \leq (d^* - 1)/2 - s(\mathcal{X}_q)$, we have shown moreover, that every solution of the modified key equation is also an error-locator polynomial. Example 4.7.(b) below shows that this is not true if we omit this assumption. (Similar examples can also be found in [19, Section VIII] and [4, Section 1.4.6].)

Proposition 4.5.(a) allows in many cases to recognize whether solutions of the modified key equation are error-locator polynomials or not. To formulate it, we need to introduce some notation. At the end of this section, we translate this proposition into a practical algorithm.

Let $(\Lambda, \tilde{R})$ be a solution of the modified key equation. Consider the set $\mathcal{Z}_\Lambda \subset \mathcal{X}_q(\mathbb{F}_{q^2}) \setminus \{P_0, P\}$ of poles of $\tilde{R}/\Lambda$ different from P, P_0. Note that $\mathcal{Z}_\Lambda$ is a subset of the set of $\mathbb{F}_{q^2}$-rational zeros of Λ. For $P_i \in \mathcal{Z}_\Lambda$, we write

$$-\frac{e_{\Lambda,i}\beta_i^{b_m+1}}{x - \alpha_i} \tag{10}$$

for the polar part of $\tilde{R}/\Lambda$ in $P_i = (\alpha_i, \beta_i)$. For $i \notin I \cup \{0\}$ we set $e_i = 0$. For P_0, define

$$e_{\Lambda,0} = \begin{cases} s_{0,0} - \sum_{j=1}^{n-1} e_{\Lambda,j} & \text{if } P_0 \text{ is a zero of } \Lambda \text{ ,} \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

Denote by $t(\Lambda)$ the degree of $\mathcal{Z}_\Lambda$ if $e_0 = 0$ and the degree of $\mathcal{Z}_\Lambda + 1$ otherwise. We write $\boldsymbol{e}_\Lambda = (e_{\Lambda,i})$.

Note that $\tilde{R}/\Lambda$ does not have poles outside $\mathcal{X}_q(\mathbb{F}_{q^2})$ since $\tilde{R} - \Lambda U \in L((\mu + \ell)P + Q - (q+1)(b_m+1)P_0)$ (Proposition 3.1). This implies that $t(\Lambda)$ is less than or equal to the number of $\mathbb{F}_{q^2}$-rational zeros of Λ.

Proposition 4.5 *Suppose that $(\Lambda, \tilde{R})$ is a solution of the modified key equation, such that $\mu := \rho(\Lambda) < d^*$.*

(a) *If Λ is the minimal error-locator polynomial then $\mu \leq N$, where N is minimal such that $\ell(NP) > t(\Lambda)$. In particular $\mu \leq t(\Lambda) + g$.*

(b) *For $\boldsymbol{e}_\Lambda$ as above, we have that Λ is an error-locator polynomial if and only if $\boldsymbol{r} - \boldsymbol{e}_\Lambda$ is a codeword. (Recall that $\boldsymbol{r}$ denotes the received vector.).*

Proof: The statement of (a) follows immediately from Lemma 2.6.

The forward direction of (b) was already shown in Proposition 3.3. For the backwards direction, we assume that $\boldsymbol{c}_\Lambda := \boldsymbol{r} - \boldsymbol{e}_\Lambda$ is a codeword. The definition of $e_{\Lambda,i}$ immediately implies that $\Lambda(P_i) = 0$ for all i. $\square$

Remark 4.6 Suppose it is known that the number t of errors satisfies $t \leq (d_{\min}-1)/2$. Then the performance of the algorithm improves if one also removes solutions with $t(\Lambda) > (d_{\min} - 1)/2$.

Example 4.7 Let $q = 4$ and $n = 64$, i.e. $\{P_0, \ldots, P_{63}\} = \mathcal{X}_q(\mathbb{F}_{q^2}) \setminus \{P\}$. We consider the code corresponding to the Riemann–Roch space $L(mP)$ with $m = 51$. One computes that $\ell = 12$, $m^\perp = 23$, $b_m = 4$ and $d^* = d_{\min} = 13$ (see [10, Section 5.3]). We represent $\mathbb{F}_{16} \simeq \mathbb{F}_2[x]/(x^4 + x + 1)$. Let $\alpha \in \mathbb{F}_{16}$ be a zero of $x^4 + x + 1 = 0$. Note that $\alpha \in \mathbb{F}_{16}^*$ is an element of order 15. Suppose that $t = 6 = (d_{\min} - 1)/2$.

(a) We first consider an example where P_0 is among the error positions. We choose error positions

$$P_1 = (1, \alpha), P_2 = (1, \alpha^2), P_3 = (\alpha^8, \alpha^{14}), P_4 = (1, \alpha^8), P_5 = (\alpha, \alpha^6), P_0 = (0, 0)$$

and error values $e_i = 1$ for $i = 1, \ldots, 6$.

The modified syndrome polynomial is

$$\begin{aligned}\tilde{S} = \alpha^5 x^3 y^4 + &\alpha^5 x^4 y^3 + \alpha^{10} x^2 y^4 + \alpha^4 x^3 y^3 + \alpha^{10} x^4 y^2 + \alpha^4 x y^4 + \alpha^{10} x^2 y^3 \\ &+ \alpha^2 x^3 y^2 + \alpha^{14} x^4 y + \alpha^5 y^4 + \alpha^6 x y^3 + \alpha^8 x^2 y^2 + \alpha^7 x^3 y + \alpha^5 x^4 \\ &+ \alpha^5 y^3 + \alpha^{12} x y^2 + \alpha^8 x^2 y.\end{aligned}$$

The minimal solution $(\Lambda, \tilde{R})$ of the modified key equation satisfies $\rho(\Lambda) = 12$. We have that

$$\begin{aligned}\Lambda &= x^3 + \alpha x y + \alpha^{12} x^2 + \alpha y + \alpha^{11} x = \alpha(x+1)(\alpha^{14} x^2 + y + \alpha^{10} x), \\ \tilde{R} &= \alpha^5 x y^4 + \cdots + \alpha y.\end{aligned}$$

The $\mathbb{F}_{q^2}$-rational zeros of Λ are the error positions, together with $(1, \alpha^4)$. The other 5 zeros of Λ are not $\mathbb{F}_{q^2}$-rational, and hence certainly not poles of $\tilde{R}/\Lambda$. This polynomial satisfies the criterion of Proposition 4.5.(a).

By computing the polar part of $\tilde{R}/\Lambda$, one finds that $e_i = 1$ for $i \neq 0$. Note that $\tilde{R}/\Lambda$ does not have a pole in P_0. As already remarked in Section 2, we need to calculate the error value in P_0 a bit differently. Since $s_{0,0} = 0 \neq \sum_{j=1}^{5} e_j$, we conclude that P_0 is an error position. Moreover, we conclude that

$$e_0 = \sum_{j=1}^{5} e_j = 1.$$

(b) Suppose that the error positions are

$$P_1 = (1, \alpha), P_2 = (1, \alpha^2), P_3 = (1, \alpha^4), P_4 = (1, \alpha^8), P_5 = (\alpha, \alpha^6), P_6 = (\alpha^2, \alpha^3)$$

and the error values $e_i = 1$ for $i = 1, \ldots, 6$. One computes that the modified syndrome polynomial is

$$\begin{aligned}\tilde{S} = \alpha^5 x^3 y^4 + \alpha^2 x^4 y^3 + \alpha^{10} x^2 y^4 + \alpha^{13} x^3 y^3 + \alpha^4 x^4 y^2 + \alpha^2 x y^4 \\ + \alpha^{11} x^2 y^3 + \alpha^4 x^4 y + \alpha^5 y^4 + \alpha^{11} x^2 y^2 + \alpha^6 x^3 y + \alpha^8 x^4 \\ + \alpha^{14} y^3 + \alpha^{11} x y^2 + \alpha^9 x^2 y.\end{aligned}$$

The minimal solution (Λ, R) of the modified key equation satisfies $\rho(\Lambda) = 9$; a solution is for example given by

$$\begin{aligned}\Delta_4 &= xy + \alpha^5 x^2 + y + \alpha^{13} x + \alpha^8, \\ R_4 &= \alpha y^4 + \text{ lower terms.}\end{aligned}$$

(This can for example be computed with the algorithm of the next section. The terminology Δ_i, R_i will also be explained there.) Therefore a minimal error-locator polynomial satisfies $\rho(\Lambda) \geq 9$.

Note that Δ_4 has exactly three $\mathbb{F}_{q^2}$-rational zeros. Since $\ell(8P) > 3$, the criterion of Proposition 4.5.(a) implies that Δ_4 is not an error-locator polynomial (we do not even need to consider R_4 to check this). This is no contradiction to Lemma 4.3, since $t = (d_{\min} - 1)/2 > (d_{\min} - 1)/2 - s(\mathcal{X}_q)$.

We now explain how to check the statement of Proposition 4.5.(b) in practice. For this, we define

$$V := \sum_{i: P_i \neq (0,0)} e_{\Lambda, i} u_i,$$

and

$$G := \frac{\tilde{R}}{\Lambda} - V. \qquad (12)$$

As in the proof of Proposition 3.3, one shows that $G \in L(\ell P)$. We may compute the coefficients of $G = \sum G_i \varphi_i$, for example by substituting suitable points P_i in (12).

We first suppose that Λ is an error-locator polynomial with $\rho(\Lambda) < d^*$. Let (Λ, R) be the solution of the key equation corresponding to Λ. Then it follows from Proposition 3.3 that $V = U = R/\Lambda$. Therefore

$$G = \frac{\tilde{R}}{\Lambda} - \frac{R}{\Lambda} = \tilde{S} - S.$$

We conclude that if Λ is an error-locator polynomial, then the coefficients of G are precisely the unknown syndromes.

Now suppose that $(\Lambda, \tilde{R})$ is any solution of the modified key equation. Recall that for $j \neq 0$ we computed the values $e_j = e_{\Lambda,j}$ via the polar part of $\tilde{R}/\Lambda$. For $0 \leq \rho(\varphi_i) \leq \ell$, we write $\varphi_i = x^{a_m - a} y^{b_m - b}$ if such a monomial exists in $\mathcal{R}$. We may now check whether the coefficients G_i satisfy

$$G_i = s_{a,b} = \sum_{j=0}^{n} e_{\Lambda,j}\varphi_{a,b}(P_j). \tag{13}$$

Example 4.8 We apply this procedure to the solution (Δ_4, R_4) of Example 4.7.(b). (Recall that we have in fact already shown that Δ_4 is not an error-locator polynomial.) We first compute the polar parts of R_4/Δ_4 in the three $\mathbb{F}_{q^2}$-rational zeros of Δ_4 which we denote by $\{P_1, P_2, P_3\}$. Since R_4/Δ_4 has a simple pole in these three points $P_i = (\alpha_i, \beta_i)$, the polar part in P_i is a multiple of $1/(x - \alpha_i)$. These “residues” satisfy:

$$e_1 := \mathrm{Res}_{(\alpha,\alpha^6)} \frac{R_4}{\Delta_4} = \alpha^{10}, e_2 := \mathrm{Res}_{(\alpha^5,\alpha^{11})} \frac{R_4}{\Delta_4} = \alpha^{11}, e_3 := \mathrm{Res}_{(\alpha^2,\alpha^3)} \frac{R_4}{\Delta_4} = \alpha^2.$$

From this we compute that the polynomial G defined by (12) is given by

$$G = \alpha^{14} + \alpha^{13}x + \alpha^{12}y + x^2 + \alpha^{14}y^2.$$

This can for example be computed by substituting $x = \alpha, \alpha^2, \alpha^5$ and simplifying the obtained rational functions in one variable.

One checks that

$$G_0 = \alpha^{14} \neq s_{4,4} = \sum_{j=1}^{3} e_j \varphi_{4,4}(P_j) = \alpha^{12}.$$

This gives a second proof that Δ_4 is not an error-locator polynomial.

The following algorithm summarizes our criterion for recognizing error-locator polynomials as follows. Assume we are given a solution (Λ, R) of the modified key equation such that $\mu := \rho(\Lambda) < d_{\min}$. Since we only need to find the minimal error-locator polynomial, we may assume that no error-locator polynomial with $\rho(\Lambda) < \mu$ exists. Assume moreover that the number t of errors is less than or equal to $(d_{\min} - 1)/2$. (Compare to Remark 4.6.)

Algorithm 4.9 Input: μ, together with several solutions (Λ, R) of the modified key equation with $\rho(\Lambda) = \mu$. Output: an error-locator polynomial. If (d*) is applied instead of (d), one obtains a (smaller) list of candidates for the error-locator polynomial, which will mostly consist of one element.

(a) Determine the $\mathbb{F}_{q^2}$-rational zeros t_0 of Λ. Let N_0 be minimal such that $\ell(N_0 P) > t_0$. If $\mu > N_0$ then reject Λ.

(b) Otherwise, determine $e_{\Lambda,i}$ for $\Lambda(P_i) = 0$ and $i \neq 0$ by (10). Determine $e_{\Lambda,0}$ by (11).

(c) Let $t(\Lambda)$ be the number of i for which e_i is nonzero, and compute N as in the statement of Proposition 4.5.(a). If $\mu > N$ or $t(\Lambda) > (d_{\min} - 1)/2$ then reject Λ.

(d) In case one has found more than one different candidate error-locator polynomials Λ_i of the same degree, apply the criterion of Proposition 4.5.(b): compute $\boldsymbol{c}_{\Lambda_i} := \boldsymbol{r} - \boldsymbol{e}_{\Lambda_i}$ and determine for which i the word $\boldsymbol{c}_{\Lambda_i}$ is a codeword.

(d*) This step is an alternative to step (d) in the case that several candidates are found. Choose the solution Λ for which the number $t(\Lambda)$ is maximal, if it is unique.

Unfortunately, Step (d) of the above algorithm increases the running time of the algorithm (Section 6). Simulation results show that this has to be invoked only in a very small number of cases (see Section 5.1). Applying the alternative step (d*) in stead of (d) works almost as well and is faster. The optimal version seems to be to combine steps (d) and (d*) as follows: apply (d*) to see if this yields a unique answer. Only apply (d) in the case that there are several solutions Λ_i of the key equation with the same degree and the same number of $\mathbb{F}_{q^2}$-rational zeros such that $e_{\Lambda,i} \neq 0$. An example of this very rare phenomenon is given in Example 5.5.

Remark 4.10 We now explain why (d*) is a reasonable choice. Assume that μ is the degree of a minimal error-locator polynomial of some code. Consider all polynomials $\Lambda \in L(\mu P)$. Then the error-locator will have much more than average many $\mathbb{F}_{q^2}$-rational points.

As a second step, one recalls that, ignoring P_0, the error are $\mathbb{F}_{q^2}$-rational poles of R/Λ. Zeros of Λ which are not poles of R/Λ may therefore be ignored. Note moreover that we have shown that zeros of Λ which are not $\mathbb{F}_{q^2}$-rational are never poles of R/Λ.

We have tested this approach on many examples, see Section 5.1 for a summary of these simulations.

5 Calculating the minimal error-locator polynomial.

In this section we explain a practical algorithm for solving the modified key equation. The algorithm explained here is an extension of a modification of the subresultant-sequence algorithm introduced by Shen [22]. The main difference between the two algorithms is that Shen's algorithm uses matrix operations to calculate a subresultant sequence which is known to yield the same polynomials (up to a constant factor) as the Euclidean algorithm for univariate polynomials.

In contrast, our algorithm mimics the steps of the Sugiyama algorithm used for decoding RS-codes more closely. Further, our algorithm easily extends to the decoding of most error patterns of weight $\lfloor (d_{\min} - 1)/2 \rfloor$; such an extension is not given in [22].

The core part of the algorithm is the computation of a new basis Δ_i of $L(\mu P)$, together with polynomials R_i that satisfy the first part of the modified key equation, i.e.

$$\operatorname{ord}_{P_0}(\tilde{S}\Delta_i - R_i) \geq (q+1)(b_m+1).$$

Here μ is a bound that will be adapted as we go along. The (Δ_i, R_i) are defined in such a way that $\rho(\Delta_i) = \rho(\varphi_i)$ and moreover the degree of R_i is minimal. Certain additional choices are made to make the polynomials uniquely determined. The iterative step of our algorithm (Step 1) mimics the division algorithm as used in the Groebner-basis algorithm. We never actually compute any Groebner basis. The degree function ρ plays the role of the ordering in the Groebner-basis setting. A summary of our algorithm in pseudocode can be found in [14].

For an introduction to the use of Groebner bases in coding theory, we refer to [3, Chapter 9], [2].

1. Compute (Δ_i, R_i) **until** $\rho(R_i) - \rho(\Delta_i) \leq \ell$

The computation is based on the division of a bivariate polynomial by j other bivariate polynomials. In such a division, j quotient polynomials $\gamma_1, \ldots, \gamma_j$ and one remainder polynomial are obtained by repeatedly subtracting (multiples of) the j divisor polynomials until no term in the dividend is a multiple of the leading monomial of any divisor.

Since the first part of the modified key equation is a statement "modulo y^{b_m+1}", we will compute modulo this relation. Concretely, if we write $f \equiv g \pmod{y^{b_m+1}}$, we mean that $\operatorname{ord}_{P_0}(f-g) \geq \operatorname{ord}_{P_0}(y^{b_m+1}) = (q+1)(b_m+1)$ in the ring $\mathcal{R}$. Similarly, by $[f]$ we denote the unique polynomial with $[f] \equiv f \pmod{y^{b_m+1}}$ which does not contain any monomials $x^a y^b$ with $\operatorname{ord}_{P_0}(x^a y^b) \geq (q+1)(b_m+1)$. (This is equivalent to $b \geq b_m$.)

1.a Initialization We set $i = 0$ and $\Delta_0(x,y) = 1$, $R_0(x,y) = \tilde{S}(x,y)$. Note that the first part of the modified key equation (9) is trivially fulfilled. The second part is only fulfilled if all syndrome elements are zero which is equivalent to the received word being a codeword. In this case, no decoding is neccessary, and we may stop the algorithm here. Otherwise, continue with the next step.

1.b Computing the quotients polynomials $\gamma_{i,j}$ Raise i by 1. Define

$$\varphi_{i_1} := \begin{cases} x^{a-1} & \text{if } \varphi_i = x^a, \\ \varphi_i / y & \text{otherwise.} \end{cases}$$

Put $\theta = [xR_{i_1}]$ if $\varphi_{i_1} = x^{a-1}$ and $\theta = [yR_{i_1}]$ otherwise. In the following, we restrict to the case that $\theta = [yR_{i_1}]$. The other case is completely analogous.

Let $\gamma_{i,j}$ be the quotients $\pmod{y^{b_m+1}}$ of dividing θ by $R_{i-1}, R_{i-2}, \ldots, R_0$ (in that order) imposing the additional condition

$$\rho(\Delta_j) + \rho(\gamma_{i,j}) < \rho(\varphi_i). \tag{14}$$

Let R_i be the remainder $\pmod{y^{b_m+1}}$ of this division. This means that we may write

$$\theta = \sum_{j<i} \gamma_{i,j} R_j + R_i.$$

Note that the polynomials $\gamma_{i,j}$ and R_i are uniquely determined by these requirements. Moreover, it follows that the R_i have distinct degree.

We already know that $R_j = \Delta_j \tilde{S} \pmod{y^{b_m+1}}$ for all $j < i$. Therefore it follows from the definition $\theta = yR_{i_1}$ that

$$y\Delta_{i_1}\tilde{S} = \sum_{j<i} \gamma_{i,j} \Delta_j \tilde{S} + R_i \pmod{y^{b_m+1}}.$$

We define $\Delta_i = y\Delta_{i_1} - \sum_{j<i} \gamma_{i,j}\Delta_j$. It follows that (Δ_i, R_i) fulfills the first part of (9). Equation (14) implies that $\rho(\Delta_i) = \rho(\varphi_i)$. In particular, $\Delta_0, \ldots, \Delta_i$ form a basis of $L(\mu_i P)$, for $\mu_i = \rho(\Delta_i)$.

1c: Termination In each step we compute $\rho(R_i) - \rho(\Delta_i)$, and stop as soon as this number is less than or equal to ℓ. Moreover, we have seen that the degree $\rho(\Lambda)$ of a minimal error-locator polynomial is less than $t + g$. This gives an upper bound on the number of cycles we need.

Example 5.1 We illustrate the division procedure in Example 4.7.(b). Since the expressions for R_i are rather long, we only give the main terms.

Initialization: Set $i = 0, \Delta_0 = 1$,

$$R_0 = \tilde{S} = \alpha^5 x^3 y^4 + \alpha^2 x^4 y^3 + \alpha^{10} x^2 y^4 + \alpha^{13} x^3 y^3 + \alpha^4 x^4 y^2 + \alpha^2 x y^4 + \alpha^{11} x^2 y^3 \\ + \alpha^3 x^3 y^2 + \alpha^4 x^4 y + \cdots + \alpha^9 x^2 y.$$

The complete expression for $R_0 = \tilde{R}$ can be found in Example 4.7.(b).

The step $i = 1$: We have $\varphi_1 = x = x \cdot 1$, hence $i_1 = 0$. Therefore

$$\theta = [xR_0] = \alpha^5 x^4 y^4 + \alpha^2 y^4 + \cdots .$$

Note that we compute modulo $y^{b_m+1} = y^5$ in $\mathcal{R}$. Dividing θ by R_0, under the additional condition that $\rho(\gamma_{1,0}) < \rho(\varphi_1) - \rho(\delta_0) = 4$, has quotient $\gamma_{1,0} = 0$. It follows that $R_1 = \theta$ and $\Delta_1 = x\Delta_0 = x$.

The step $i = 2$: We have that $\varphi_2 = y = y \cdot 1$, hence $i_1 = 1$. Therefore

$$\theta = [yR_0] = \alpha^2 x^4 y^4 + \alpha^{13} x^3 y^4 + \alpha^4 x^4 y^3 + \cdots .$$

We divide θ first by R_1. As before, we find quotient $\gamma_{2,1} = \alpha^{12}$ and remainder $\varepsilon := \alpha^5 x^3 y^4 + \alpha^2 x^4 y^3 + \cdots$. Dividing ε by R_0 yields the quotient $\gamma_{2,0} = 1$ and the remainder $R_2 = x^4 y^2 + \cdots$. This yields $\Delta_2 = y\Delta_0 - \gamma_{2,1}\Delta_1 - \gamma_{2,0}\Delta_0 = y + \alpha^{12} x + 1$. A tabular summary of the rest of the algorithm can be found in Example 5.5 below.

Remark 5.2 Simulations have shown that our algorithm works just as well if we stop each iteration as soon as one term in θ could not be canceled instead of performing the entire division. A justification for this is that $\rho(R_i)$ does not change with the modified division, and the minimality of the returned solution depends on $\rho(R_i)$ only, not on terms of smaller order. In the case that $t \leq \lfloor (d_{\min} - 1)/2 \rfloor$, the uniqueness of the minimal solution follows from the theory of Riemann-Roch spaces (Corollary 3.2).

Lemma 5.3 *Let $\Lambda \in L(\mu_i P) \setminus L((\mu_i - 1)P)$. Write $\Lambda = \sum_{j=0}^{i} c_j \Delta_j$ with $c_j \in \mathbb{F}_{q^2}$. Put $R = \sum_{j=0}^{i} c_j R_j$. Then $\operatorname{ord}_{P_0}(\Lambda \tilde{S} - R) \geq (q+1)(b_m + 1)$. Moreover, R is the unique polynomial with this property such that $\rho(R) - \rho(\Lambda)$ is minimal, for given Λ.*

Proof: Let Λ, R be as in the statement of the lemma. Recall that the Δ_j form a basis of $L(\mu_i P)$, therefore there exist unique c_j as in the statement of the lemma. The properties of (Δ_j, R_j) imply that $\operatorname{ord}_{P_0}(\Delta_j \tilde{S} - R_j) \geq (q+1)(b_m+1)$. Therefore the same property holds for (Λ, R). The uniqueness of R follows from Remark 4.2. $\Box$

The following theorem follows immediately from Lemma 5.3. A *minimal solution to the modified key equation* is a solution (Λ, R) such that $\rho(\Lambda)$ is minimal.

Theorem 5.4 *Step 1 of the algorithm computes a minimal solution to the key equation.*

Proof: Our algorithm constructs a series of pairs (Δ_i, R_i), where every possible $\rho(\Delta_i)$ is obtained in increasing order. The algorithm stops as soon as the second part of (3) is fulfilled for (Δ_i, R_i). Let i be the value for which the algorithm terminates, and let $\mu = \rho(\Lambda)$ be the degree of a minimal solution of the modified key equation. To prove the algorithm, it suffices to show that $\mu = \mu_i$.

Assume that $\mu = \rho(\Lambda)$ is the degree of a minimal solution (Λ, R) of the modified key equation. Choose j such that $\mu_j = \mu$. We need to show that (Δ_j, R_j) is a solution of the modified key equation. Since $\Delta_0, \ldots, \Delta_j$ form a basis of $L(\mu P)$, we may write $\Lambda = \sum_{s \leq j} c_s \Lambda_s$. Recall that $R = \sum_{s \leq j} c_s R_s$. Since $\rho(\Lambda) = \rho(\Delta_j) = \mu_j$, we have $c_j \neq 0$.

Assume that (Δ_j, R_j) is not a solution to the modified key equation. Then $\rho(R) \leq \ell + \mu$ and $\rho(R_j) > \ell + \mu$. In particular, we have that $\rho(R) < \rho(R_j)$. But this contradicts $R = \sum_{s \leq j} c_s R_s$ with $c_j \neq 0$, since by construction the polynomials R_s have different degree. $\Box$

In the rest of this section, we explain how to use Theorem 5.4 to compute a minimal error-locator polynomial.

2. Obtaining the error-locator polynomial

From the previous step, we obtain (Δ_i, R_i) with $\rho(R_i) - \rho(\Delta_i) \leq \ell$. Put $\mu_i =$

$\rho(\Delta_i)$. If there exists an error-locator polynomial Λ with $\rho(\Lambda) \leq \mu_i$ then we may write

$$\Lambda = \sum_{j \leq i} c_j \Delta_j, \tag{15}$$

since the Δ_i form a basis of $L(\mu_i P)$. Recall that the corresponding polynomial R such that (Λ, R) is a solution of the modified key equation can then be written as

$$R = \sum_{j \leq i} c_j R_j.$$

The division algorithm implies that the $\rho(R_j)$ are all distinct. Therefore for all j with $c_j = 0$ we have that

$$\rho(R_j) - \rho(\Delta_i) > \ell.$$

This greatly limits the linear combinations to consider.

In the case that $\Lambda = \Delta_i$ (i.e. $c_j = 0$ for all $j < i$) we have defined $R = R_i$. Therefore we need to check whether (Λ, R) is an error-locator polynomial using Algorithm 4.9, which is particularly easy since we only have one candidate. Namely, it suffices to check whether $t(\Lambda)$ and $\mu_i = \rho(\Delta_i)$ satisfies the concrete bounds in that statement.

In step 3 we explain what to do if Λ is not an error-locator polynomial by the criteria of Algorithm 4.9. The general case that $c_j \neq 0$ for at least one $j < i$ is treated in step 4.

3. Treatment of decoding failures
If no error locator is found by the previous basis, increase μ and i to include the next (Δ_i, R_i) which is a solution to the key equation, and repeat step 2. If we can bound the number of errors (usually we want $t \leq \lfloor (d_{\min} - 1)/2 \rfloor$), then by Lemma 2.6 $\rho(\Lambda) \leq t + g$, so we know that it suffices to run the algorithm until $\rho(\Delta_i)$ reaches this bound, hence the number of additional steps is limited. If no further pair (Δ_i, R_i) fulfilling the key equation is found, we know that more than t errors occurred. In our setting this means that the error weight exceeds half the minimum distance and unique decoding cannot be guaranteed any more, making a special treatment necessary. More details about this situation can be found in Section 7.

4. Choosing a single solution and computing the error values
As mentioned in step 2, we might find more than one possible solution to the key equation that also is an error locator. This problem has been dealt with in Algorithm 4.9. Note that in this part of the algorithm, we also compute the error values.

Example 5.5 We continue with Example 4.7.(b), and illustrate the different steps of the above algorithm. Since the expressions for R_i are rather long, we omit them.

i	Δ_i	$\rho(\Delta_i)$	$\rho(R_i)$
0	1	0	32
1	x	4	36
2	$y+\alpha^{12}x+1$	5	26
3	$x^2+\alpha^5x+\alpha^3$	8	21
4	$xy+\alpha^5x^2+y+\alpha^{13}x+\alpha^8$	9	20

We see that $i = 4$ is the first value such that $\rho(\Delta_i) - \rho(R_i) \leq \ell = 12$. This implies immediately that the minimal error-locator polynomial has degree greater than or equal to 9. Moreover, we have already checked that Δ_4 is not an error-locator polynomial (Example 4.7.(b)). Note that $\rho(R_3) - \rho(\Delta_4) = 12$ but that for $i = 0, 1, 2$ we have that $\rho(R_i) - \rho(\Delta_4) > 12$. This implies that an error-locator polynomial Λ of degree 9, if it exists, may be written as

$$\Lambda = \Delta_4 + c\Delta_3, \qquad \text{with } c \in \mathbb{F}_{16}.$$

Checking the criterion of Proposition 3.3 for all values of c yields as candidates

$$\begin{aligned}
\Lambda_1 &:= \Delta_4 + \alpha^{14}\Delta_3 = (x-1)(y+\alpha^{12}x+1),\\
\Lambda_2 &:= \Delta_4 + \Delta_3 = xy + \alpha^{10}x^2 + y + \alpha^7 x + \alpha^{13} \qquad \text{and}\\
\Lambda_3 &:= \Delta_4 + \alpha^6\Delta_3 = xy + \alpha^9 x^2 + y + \alpha^4 x + \alpha^{12}
\end{aligned}$$

satisfying (a) of Algorithm 4.9, since the value $t_0(\Lambda_i)$ of $\mathbb{F}_{q^2}$-rational zeros of these polynomials is $t_0(\Lambda_1) = t_0(\Lambda_2) = 9$ and $t_0(\Lambda_3) = 6$.

We now apply the other steps of Algorithm 4.9. We write $\tilde{R}_i$ for the unique polynomial as in Lemma 5.3. One computes that $t(\Lambda_i)$ (defined in Algorithm 4.9.(c)) satisfies: $t(\Lambda_1) = t(\Lambda_3) = 6$ and $t(\Lambda_2) = 9$. Since $t(\Lambda_2) > (d_{\min} - 1)/2$, this solution is rejected by Algorithm 4.9.(c).

Since $t(\Lambda_1) = t(\Lambda_3)$, criterion (d*) does not distinguish between the candidates Λ_1 and Λ_3, therefore we apply (d). We compute the error values $\boldsymbol{e}_{\Lambda_3}$, and find:

$$e_{\Lambda_3,(\alpha,\alpha^6)} = \alpha^9, \quad e_{\Lambda_3,(\alpha^2,\alpha^3)} = \alpha^6, \quad e_{\Lambda_3,(\alpha^6,\alpha^4)} = 1, \quad e_{\Lambda_3,(\alpha^{10},\alpha^7)} = \alpha,$$
$$e_{\Lambda_3,(\alpha^{13},\alpha^7)} = \alpha^6, \quad e_{\Lambda_3,(\alpha^{14},\alpha^3)} = \alpha.$$

One checks for example that $s_{1,0} = \alpha^5 \neq \sum_j e_j\varphi_{1,0}(P_j) = \alpha$. We conclude that Λ_3 is not an error-locator polynomial, and therefore select Λ_1. Alternatively, one may also check that the criterion of Algorithm 4.9.(d) is satisfied for Λ_1 which immediately proves that Λ_1 is the minimal error-locator polynomial.

The method described in this section does not only work for the minimal error-locator polynomial. Continuing the iterations, one finds for example

$$\Lambda_4 := \Delta_6 + \alpha^2\Delta_4 = (x-1)(x^2 + \alpha^{10}y + \alpha^{13}x + \alpha^{12}),$$

which is a further error-locator polynomial corresponding to the same codeword.

Since all common zeros of Λ_1 and Λ_4 on $\mathcal{X}_q$ are error locations, it follows that (Λ_1, Λ_4) is a set of generators of the error-locator ideal. (Alternatively, one may also check that $\dim_{\mathbb{F}_q} \mathcal{R}/\langle \Lambda_1, \Lambda_4 \rangle = t = 6$.)

We compare this to the method of [18]. Their algorithm needs 20 rounds to compute a set of generators of the error-locator ideal I_e in this particular example. In practice, the minimal generators are already computed in earlier steps, but the algorithm does not check this.

What remains to complete the decoding process is to find the error values. But once the error locator polynomial has been determined this is a relatively simple task. Basically, two different possibilities exist: the first is to use the error-locator polynomial to recursively extend the syndrome polynomial until the error values can be found by evaluation. Such an approach is described e.g. in [21]. The other possibility is to exploit Lemma 4.3 and calculate the residues as was done in Example 4.8 to obtain the error values. Note that the latter approach is similar to using the well-known Forney formula for RS codes [20, p. 195], and had also been presented in other works, e.g. [5], [19].

5.1 Simulation results We describe several simulation results, based on an implementation of our algorithm in MAGMA. The main goal of these simulations is to illustrate the effectivity of step (d*) of Algorithm 4.9. Therefore, we have consistently used (d*) rather than (d). In the case that (d*) does not yield a unique solution, we have randomly chosen one of the candidates. Surprisingly enough, this yields very often the correct results, especially for codes with small rates. Recall that using (d) instead always yields the correct codeword, since we assume that $t \leq \lfloor (d_{\min} - 1)/2 \rfloor$.

Table 1 presents the results of a series of simulations that was performed for Hermite codes with several design parameters m over $\mathbb{F}_{4^2}$. The choice $q = 4$ yields $s = 1$, so it is only necessary to simulate the decoding of errors with weight $t = \lfloor (d_{\min} - 1)/2 \rfloor$. In case we find multiple solutions (Λ_i, R_i) of the modified key equation with equal value $\rho(\Lambda_i)$, we use (d*) of Algorithm 4.9 in stead of (d), and select those solutions Λ_i for which the number $t(\Lambda_i)$ of poles of R_i/Λ_i is maximal. If Λ_i is still not unique, we make a random choice.

The design parameters, code rates k/n and tested error weight are given in the first three columns of Table 1. For each code, 10^7 random error patterns were used. The number E_f in the fourth column gives the number of error patterns for which the first solution returned by the algorithm was erroneously accepted. The number N_b shows the number of error words for which a second basis polynomial was calculated, and the number E_b denotes the number of errors where the criterion led to a wrong decision among the candidates.

It can be seen that for small code rates correct decoding is already achieved, but small error rates remain for codes with higher rates.

m	$\frac{k}{n}$	$\left\lfloor\frac{d_{\min}-1}{2}\right\rfloor$	E_f	N_b	E_b
27	0.344	18	0	2034	0
33	0.438	15	0	2050	0
37	0.5	13	1	2170	1
43	0.563	10	7	2018	1
47	0.656	8	572	3150	108

Table 1: Simulation Results for Several Codes $H(m)$

6 The complexity of the division algorithm

In this section, we first prove that the complexity of the algorithm presented before is at most cubic. This is worse than the complexity of the fastest known algorithms, yet analysis of simulations showed that the practical asymptotic complexity is $\mathcal{O}(n^{7/3})$, which is the same as that of other common decoding algorithms for AG-codes.

6.1 The worst case complexity

Lemma 6.1 *Assume that $t \leq \lfloor (d_{\min} - 1)/2 \rfloor - s(\mathcal{X}_q)$ errors occurred. Finding a minimal solution to the key equation with the algorithm presented in the previous section has complexity $\mathcal{O}(n^3)$.*

Proof: The main part of the decoding algorithm is the determination of an error-locator polynomial. To estimate its complexity, we count the number of necessary multiplications in $\mathbb{F}_{q^2}$. Throughout this section, we use the notations introduced in Section 5, and let $\tau = \lfloor (d_{\min} - 1)/2 \rfloor - s(\mathcal{X}_q)$ be the maximum number of correctable errors. Any error of weight $t \leq \tau$ can then be decoded with at most the same complexity. Because the check matrix can be precalculated, the computation of the syndrome polynomial has complexity $\mathcal{O}(nm^{\perp})$. The selection of i_1 has linear complexity, and the calculation of θ has $\mathcal{O}(m^{\perp})$. Next, we need to divide θ by several other polynomials. This division requires up to $\rho(\theta)\tau$ checks followed by $\rho(\theta)$ subtractions of another polynomial, where a single subtraction has complexity $\mathcal{O}(m^{\perp})$. Up to τ such divisions have to be performed, hence the overall complexity of this step is $\mathcal{O}(\rho(\theta)m^{\perp}\tau) = \mathcal{O}(n^3)$. The calculation of the polynomials Δ_i can be performed in line with the division, hence does not increase the asymptotic complexity. As stated in [5], the complexity of the evaluation step is $\mathcal{O}(n^2)$, so it does not affect the overall complexity. □

Shen's subresultant algorithm [22] also has cubic complexity. More precisely, the complexity is $\mathcal{O}((n + b_m + 1)^3)$, so our algorithm has a slightly better performance.

Unfortunately, if there is no unique minimal solution to the key equation, the complexity is dominated by selecting one of the candidates, and depends on the number of summands in (15): with every additional element, the number of polynomials to be checked according to Algorithm 4.9 increases by a factor of q^2 and theoretically the pairs from all $t+1$ operations might need to be included.

6.2 Reduced complexity in actual implementations In simulations, we found that it is always suffices to limit the number of summands in (15) to two. We implemented a version of the algorithm with the following modifications. If there are more than two summands once $\rho(R_i)-\rho(\Delta_i) \leq \ell$ is fulfilled for the first time, then the two pairs with the largest $\rho(\Delta_i)$ are chosen. If it is neccessary to compute additional iterations (as in step 3), we pick those two pairs which fulfill the stopping criterion. With this setup, only $q^2 = n^{2/3}$ linear combinations have to be considered. If the error values need to be computed (as in Algorithm 4.9.(d)) for all these candidates, the complexity of the selection step is $\mathcal{O}(n^{8/3})$. But in a combination with the simpler criterion (d*) from Algorithm 4.9, the necessary number of evaluations is mostly $\mathcal{O}(q)$, further reducing the overall complexity to $\mathcal{O}(n^{7/3})$.

On the other hand, the complexity of the bivariate division was also smaller than derived before: careful analysis showed that on average only $2(q+1)$ instead of $\rho(\theta)$ subtractions were necessary. Because $n = q^3$, this observation reduces the complexity of finding the minimal solution, and hence also the overall complexity, to $\mathcal{O}(n^{7/3})$. Algorithms of the same complexity were denoted "fast" in [12] and [21]. Note that the latter algorithm involves majority voting, so our algorithm can compete with majority voting algorithms also in terms of complexity.

7 A basis for decoding beyond half the minimum distance

In this section we discuss the abilities of our decoder to correct more than $\lfloor (d_{\min}-1)/2 \rfloor$ errors. The difference to the previous chapters is that if the minimal error locator has $LT(\Lambda) = \varphi_t$ with $t > \lfloor (d_{\min}-1)/2 \rfloor$ there will always be more than one solution to the key equation of this degree. In this section, we first describe how to modify our algorithm in order to obtain a basis for all these solutions, and then calculate the number n_b of elements in this basis. In the end, we shortly discuss why it is usually not feasible to use this basis in decoding without further information about the error.

7.1 The Modification to the Algorithm The original description of our algorithm uses a stopping criterion based on an upper bound on the number of errors in the received word. Unfortunately, we cannot use this stopping criterion if $t > \lfloor (d_{\min}-1)/2 \rfloor$. Instead, we use a modification of the algorithm that provides us with a basis for all solutions to the key equation up to a certain degree.

The first step is to choose a number t of errors that we want to correct, so the error-locator polynomial Λ satisfies $\rho(\Lambda) \le \rho(\varphi_t)$. Note that $\rho(\varphi_t) \le m^\perp$ is required as otherwise the degree condition on R is trivially fulfilled for any polynomial of the given order. But this restriction is actually less severe than the general bound on the decoding capabilities for linear codes which require $t < d_{\mathrm{min}}$.

We calculate all pairs (Δ_i, R_i) with $\rho(\Delta_i) \le \rho(\varphi_t)$. For $\rho(\varphi_t) < d_{\mathrm{min}}$, Proposition 3.3 shows that only those pairs with $\rho(R_i) \le \rho(\varphi_t) + \ell$ may be used in a basis for all solutions to the key equation. Lemma 5.3 implies that all solutions (Λ, R) to the modified key equation with $\rho(\Lambda) \le \rho(\varphi_t)$ can be written as linear combination of the (Δ_i, R_i). If t was chosen large enough, the correct error-locator polynomial must be constructable from that basis.

7.2 The Number of Basis Pairs Now we want to count the number of basis pairs we get from this construction. In the derivation, we use the value

$$\rho_S := q^2 + (q+1)b_m$$

to estimate the degrees of the remainder polynomials throughout the iterations. Actually, ρ_S denotes the maximum possible order of the syndrome polynomial $\tilde{S}$ and $\rho_S = \rho(\tilde{S})$ if and only if $s_{0,0} \ne 0$.

Let us first assume that $\rho(R_i) = \rho_S - \rho(\varphi_i)$ for all i because this is the most common case. Then the pairs which have not been selected for the basis have

$$\rho(R_i) = \rho_S - \rho(\Delta_i) > \rho(\varphi_t) + \ell$$

or $\rho(\Delta_i) < m^\perp + 1 - \rho(\varphi_t)$. The number of basis pairs therefore equals

$$n_b = |\Phi_{\rho(\varphi_t)}| - |\Phi_{m^\perp - \rho(\varphi_t)}| = t + 1 - |\Phi_{m^\perp - \rho(\varphi_t)}|. \tag{16}$$

Assume there exists a pair $(\bar{\imath}, i)$ of indices with $\rho(R_i) = \rho_S - \rho(\varphi_{\bar{\imath}})$. Without loss of generality, we may assume that $\bar{\imath} < i$. Careful inspection of the polynomials obtained from the modified algorithm (Section 7.1) shows that then $\rho(R_{\bar{\imath}}) = \rho_S - \rho(\varphi_i)$ is also always true. We can have one of the following situations:

1. Both $\rho(R_i) \le \rho(\varphi_t) + \ell$ and $\rho(R_{\bar{\imath}}) \le \rho(\varphi_t) + \ell$, then both pairs are selected for the basis and n_b does not change. The situation is similar if both $\rho(R_i) > \rho(\varphi_t) + \ell$ and $\rho(R_{\bar{\imath}}) > \rho(\varphi_t) + \ell$, as then neither of the pairs is selected.

2. If $i < t$ and $\rho(R_i) > \rho(\varphi_t) + \ell$ but $\rho(R_{\bar{\imath}}) \le \rho(\varphi_t) + \ell$ we can again calculate n_b by (16), but now the pair $(\Delta_{\bar{\imath}}, R_{\bar{\imath}})$ is picked instead of (Δ_i, R_i).

3. If $i > t$, we obtain $\rho(R_{\bar{\imath}}) \le \rho(\varphi_t) + \ell$ for sure, so the $\bar{\imath}$th pair is included in the basis. However, the pair (Δ_i, R_i) is not even calculated because of $\rho(\Delta_i)$, so in this situation n_b is larger than indicated by (16), which turns out to be only a lower bound on the number of basis elements.

The last case occurs only rarely, in 10^7 simulations we found this case only once.

In the following special case, the general result can be simplified: let $m^\perp - \rho(\varphi_t) > 2g-2$ and $t \geq g$. This means that the number of errors is not too small, but also not too close to the minimum distance. Then we know that

$$|\Phi_{m^\perp-\rho(\varphi_t)}| = m^\perp - \rho(\varphi_t) - g + 1 = m^\perp - (t+g) - g + 1 = m^\perp - t - 2g + 1.$$

In this case, we can rewrite (16) to

$$n_b = t + 1 - (m^\perp - t - 2g + 1) = 2t - (m^\perp - 2g + 2) + 2 = 2t - d^* + 2.$$

If we further write the number of errors as $t = (d^*-1)/2 + t_0$, then

$$n_b = 2t_0 + 1.$$

This coincides with known results for RS-codes, see e.g. [13].

An open problem remains what to do now that the basis has been found. Of course we may form all possible linear combinations and then use Algorithm 4.9 to select one, but this approach has a very high complexity: even if checking the criteria had linear complexity $\mathcal{O}(n)$, the correction of two additional errors, i.e., $t_0 = 2$, leads to an overall complexity of $n^{11/3}$. This value usually increases by a factor $n^{4/3}$ for every additional error that shall be corrected. Decoding beyond half the minimum distance without such a significant increase in complexity is therefore only possible with special methods such as the use of reliability information, collaborative decoding of interleaved Hermite codes or virtual extension to an interleaved code. In [15], the latter two approaches are described in more detail and a bound on the number of errors which can be decoded with high probability using an algorithm with cubic complexity is given.

Acknowledgment

We thank Martin Bossert for suggesting the project and Martin Bossert, Vladimir Sidorenko, Alexander Zeh and Antonia Wachter for helpful discussions and comments on a previous version of this paper. We are grateful to Michael O'Sullivan for pointing out several references to us, and to the referees for helpful comments.

References

[1] P. Beelen, T. Høholdt, The decoding of algebraic geometry codes, in: *Advances in algebraic geometry codes*, Ser. Coding Theory Cryptol. 5, 99–152, 2008.

[2] D.A. Cox, J. Little, D. O'Shea, *Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra*, 3rd edition, Springer, 2007.

[3] D. Cox, J. Little, D. O'Shea, *Using algebraic geometry*, GTM 185, Springer, 1998.

[4] I.M. Duursma, Algebraic geometry codes: general theory, in: *Advances in algebraic geometry codes*, Ser. Coding Theory Cryptol. 5, 99–152, 2008.

[5] D. Ehrhard, Über das Dekodieren algebraisch-geometrischer Codes, *Dissertation, Universität Düsseldorf*, 1991.

[6] J.I. Farrán, Decoding algebraic geometry codes by the key equation, *Finite Field Appl.* 6, 207–217, 2000.

[7] G.L. Feng, T.R.N. Rao, Decoding algebraic-geometric codes up to the designed distance, *IEEE Trans. Inform. Theory* 39, 37–45, 1993.

[8] V.D. Goppa, Codes on algebraic curves, *Dokl. Akad. Nauk SSSR* 259, 1289–1290, 1981.

[9] V. Guruswami, M. Sudan, Improved decoding of Reed–Solomon and algebraic-geometric codes, *39th Annual symposium on foundations of computer science*, 1998.

[10] T. Høholdt, J.H. van Lint, R. Pellikaan, Algebraic geometry codes, in: *Handbook of coding theory*, vol 1., 871–961, 1998.

[11] T. Høholdt, J.H. van Lint, R. Pellikaan, On the decoding of algebraic-geometric codes, *IEEE Trans. Inform. Theory* 41, 1589–1614, 1995.

[12] J. Justesen, K. Larsen, H. Jensen, T. Høholdt, Fast decoding of codes from algebraic plane curves, *IEEE Trans. Inform.Theory* 38, 111–119, 1992.

[13] S. Kampf, M. Bossert, S. Bezzateev, Some results on list decoding of interleaved Reed-Solomon codes with the extended euclidean elgorithm, *Proc. Coding Theory Days in St. Petersburg 2008*, 31–36, 2008.

[14] S. Kampf, M. Bossert, I.I. Bouw, Solving the key equation for Hermitian codes with a division algorithm, *IEEE International Symposium on Information Theory*, St. Petersburg, 1008–1012, 2011.

[15] S. Kampf, Bounds on collaborative decoding of interleaved Hermitian codes with a division algorithm and virtual extension, accepted for *3ICMCTA Special Issue of Designs, Codes and Cryptography*, 2012.

[16] D.A. Leonard, A generalized Forney formula for algebraic-geometric codes, *IEEE Trans. Inform. Theory*, 42, 1263–1268, 1996.

[17] F. J. MacWilliams, N.J.A. Sloane, *The theory of error-correcting codes*, North-Holland Mathematical Library, 1988.

[18] M. O'Sullivan, M. Bras-Amorós, The key equation for one-point codes, in: *Advances in algebraic geometry codes*, Ser. Coding Theory Cryptol. 5, 99–152, 2008.

[19] S.C. Porter, B.-Z. Shen, R. Pellikaan, Decoding geometric Goppa codes using an extra place, *IEEE Trans. Inform. Theory* 38, 1663–1676, 1992.

[20] R.M. Roth, *Introduction to coding theory*, Cambridge University Press, 2006.

[21] S. Sakata, J. Justesen, Y. Madelung, H. Elbrønd, T. Høholdt, Fast decoding of algebraic-geometric codes up to the designed minimum distance, *IEEE Trans. Inform. Theory*, 41, 1672–1677, 1995.

[22] B.-Z. Shen, Solving a congruence on a graded algebra by a subresultant sequence and its application, *J. Symbolic Comput.* 14, 505–522, 1992.

[23] A. Skorobogatov, S.G. Vlăduţ, On the decoding of algebraic-geometric codes, *IEEE Trans. Inform. Theory* IT-36, 1051–1060, 1990.

[24] H. Stichenoth, *Algebraic function fields and codes*, Second edition, GTM 254, Springer-Verlag, 2009

[25] M.A. Tsfasman, S.G. Vlăduţ, *Algebraic-geometric codes*, Kluwer Academic Publishers Group, 1991.

[26] M.A. Tsfasman, S.G. Vlăduţ, T. Zink, Modular curves, Shimura curves and Goppa codes better than the Varshmov–Gilbert bound, *Math. Nachr.* 109, 21–28, 1982.

[27] K. Yang, P.V. Kumar, On the true minimal distance of Hermitian codes, in: *Coding theory and algebraic geometry* (Luminy, 1991), LNM 518, Springer, 1992.

Institute of Pure Math
Ulm University
irene.bouw@uni-ulm.de

Institute of Communications Engineering
Ulm University
sabine.kampf@uni-ulm.de