Diplomarbeit
am Fachbereich Mathematik
der Technischen Universität Darmstadt

Application of Algebraic-Geometric Codes in Cryptography

Angefertigt von
Robert Niebuhr

unter Betreuung von
Prof. Dr. Johannes Buchmann

13. Januar 2006

Acknowledgements

First of all, I would like to thank Prof. Dr. Johannes Buchmann, who made this thesis possible, and Raphael Overbeck for his continuous and most helpful advice and support.

In addition, I am very grateful to Lea Poeplau, Rafael Dahmen and my father Hans, who have taken much time to read drafts and provided me with countless corrections and suggestions. And last, but not least, I want to thank Susanne and my family for the help and encouragement I received throughout this thesis.

Contents

1 Introduction

This thesis investigates the application of algebraic-geometric codes in cryptography, with special attention to the application in the McEliece cryptosystem.

Public-key cryptosystems form an important part of cryptography. In these systems, every user has a public and a private key. The public key allows other users to encrypt messages, which can only be decoded using the secret private key. In that way, public-key cryptosystems allow easy and secure communication between all users without the need to actually meet and exchange keys.

One such system is the *McEliece Public-Key Cryptosystem*, sometimes also called McEliece Scheme. It was developed in 1987, but did not receive much attention. This is probably due to the fact that it has some disadvantages making it difficult to use in most real-world applications.

Algebraic-geometric (AG) codes are a family of linear codes first described by V.D. Goppa. Their name comes from the fact that they are defined in terms of algebraic geometry, e.g. *curves* and *divisors*. This huge family of codes contains many well-known classes of codes, for example BCH, GRS, Alternant and Goppa codes.

The original McEliece Cryptosystem uses Goppa codes. Although some classes of codes proved to be an insecure choice, we will show that algebraic-geometric codes can help to overcome some of the disadvantages mentioned earlier.

Structure of this Thesis

Chapter 2 gives an introduction to cryptography. It explains the terms necessary to understand the rest of the thesis and includes linear, cyclic and polynomial codes.

In chapter 3 we introduce the McEliece cryptosystem. We describe the way the system works and possible cryptanalytic attacks against it.

Algebraic-geometric codes are covered in chapter 4. We explain some algebraic geometry, including curves, divisors and the theorem of Riemann-Roch. Then we show how this can be used to define AG codes.

The Berlekamp-Massey-Sakata algorithm is described in chapter 5. This algorithm can be used to decode a large class of algebraic-geometric codes.

In Chapter 6 we first define a class of AG codes called Srivastava codes. Then we use them to construct a code with good parameters that can be used within the McEliece Cryptosystem. We compute the properties of the resulting cryptosystem and compare it with others, for example those using Goppa codes.

A conclusion is drawn in chapter 7 on the achieved improvements and on what has still to be done.

2 Cryptographic Background

In this section we present some cryptographic background needed to understand algebraic-geometric codes. A more detailed introduction can be found in [46].

In general, we consider words of fixed length n with letters from a finite alphabet Q. Thus words are elements of Q^n. A *code* is a subset of Q^n and the elements of the code are called *codewords*. The natural number n is the *length* of the code.

An important class of codes are linear codes. This will be the only class of codes considered in this thesis.

2.1 Linear Codes

From now on let the alphabet Q be a finite field $\mathbb{F}_q$, so $Q^n = \mathbb{F}_q^n$ is a vector space.

Definition 2.1 (Hamming distance, weight)**.** To give the 'difference' of two codewords a precise meaning the *(Hamming) distance* between two words is introduced.
Let $x, y \in \mathbb{F}_q^n$, then

$$d(x, y) = |\{i \colon x_i \neq y_i\}|.$$

The *(Hamming) weight* of a codeword is the number of non-zero entries and therefore the distance from the zero vector:

$$w(x) = |\{i \colon x_i \neq 0\}| = d(x, 0).$$

Definition 2.2. A *linear code* $\mathcal{C}$ of dimension k is a k-dimensional linear subspace of $\mathbb{F}_q^n$ and is often called an $[n, k]$ code.

The third important parameter of a code $\mathcal{C}$, besides the length and dimension, is the minimum distance between its codewords.

Definition 2.3. The *minimum Hamming distance* d of a linear code is

$$\begin{aligned} d &= \min_{u \neq v} d(u, v) \\ &= \min_{u \neq 0} w(u). \end{aligned}$$

It is often called the *minimum distance* or simply the *distance* of the code; any two codewords differ in at least d places.

A code of dimension k, length n and minimum distance d is often called an $[n, k, d]$ code.

Two types of matrices play an important role for linear codes: generator and (parity) check matrices. They are defined as follows.

Definition 2.4. If the encoding $\mathcal{E} : \mathbb{F}_q^k \to \mathbb{F}_q^n$ from message m to codeword c is done by the matrix multiplication

$$c = \mathcal{E}(m) = mG,$$

where G is a $k \times n$ matrix with entries in $\mathbb{F}_q$, then G is called *generator matrix* of the code. The rows of G form a basis of $\mathcal{C}$.

Definition 2.5. A *parity check matrix* of a linear $[n,k]$ code $\mathcal{C}$ is a $(n-k)\times n$ matrix H, such that

$$\mathcal{C} = \{x \in \mathbb{F}_q^n \colon Hx^T = 0\}.$$

Thus the rows of a check matrix generate the orthogonal complement of $\mathcal{C}$.

Example 2.6. *The binary code* $\mathcal{C} = \{0000, 0101, 1110, 1011\}$ *can be defined by a generator matrix* G*, where*

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}.$$

It can also be defined by a check matrix H *with*

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

In many cases we transmit the encoded message. Because of channel noise the received word may contain some errors, so we want to be able to at least detect or better correct these errors. Usually we do this by choosing the codeword which is closest (with respect to the Hamming metric) to the received word to minimize the probability of making a mistake.

The following lemma provides an upper bound on the number of errors a code can correct.

Lemma 2.7. *For an* $[n,k,d]$*-code* $\mathcal{C}$ *the spheres*

$$S_c = \{x \in \mathbb{F}_q^n \colon d(x,c) \le \lfloor 1/2(d-1) \rfloor\}, \quad c \in \mathcal{C},$$

do not overlap, so every received word in S_c *will be corrected to* c*. Hence this code corrects up to* $\lfloor \frac{1}{2}(d-1) \rfloor$ *errors.*

Proof. Assume two of the spheres overlap, i.e. they both contain a point $x \in \mathbb{F}_q^n$. Then the distance between the two centers of the spheres is not greater than twice the distance to x, thus not greater than $(d-1)$. This contradicts the assumption that $\mathcal{C}$ is a code with minimum distance d. □

There are some bounds on the minimum distance of a code, one of which we will explain here. First, we need the following lemma:

Lemma 2.8. *If* H *is the parity check matrix of a code of length* n*, then the code has minimum distance* d *if and only if every* $d-1$ *columns of* H *are linearly independent and some* d *columns are linearly dependent.*

Proof. There is a codeword x of weight w if and only if $Hx^T = 0$ for some vector x of weight w. Let H_i denote the i-th column of H, then this is equivalent to $\sum_{i=1}^n H_i x_i = 0$ and $w(x) = w$. This is precisely the definition for some w columns of H being linearly dependent. The minimum weight of a code is equal to its minimum distance, so we want every $d-1$ columns linearly independent, but some d columns to be linearly dependent. □

The following bound on the minimum distance is called the *Singleton bound.*

Theorem 2.9 (Singleton bound)**.** *If* $\mathcal{C}$ *is an* $[n,k,d]$ *code, then* $n-k \ge d-1$.

Proof. The rank $r = n - k$ of H is the maximum number of linearly independent columns. □

Codes with $d = n - k + 1$ are called *maximum distance seperable*, or MDS for short. The name comes from the fact that such a code has the *maximum* possible *distance* between codewords, and that the codewords may be *seperated* into message and check symbols.

For any given linear code we can construct its dual code.

Definition 2.10. If $\mathcal{C}$ is an $[n, k]$ linear code over $\mathbb{F}_q$, its *dual* or *orthogonal code* $\mathcal{C}^{\perp}$ is the set of vectors which are orthogonal to all codewords of $\mathcal{C}$:

$$\mathcal{C}^{\perp} = \{u \in \mathbb{F}_q^n \colon u^t v = 0 \text{ for all } v \in \mathcal{C}\}.$$

With these definitions we are able to define some basic problems of coding theory.

2.2 Problems of Coding Theory

The general decoding problem for linear codes is defined as follows:

Problem 2.11. *Let $\mathcal{C}$ be an $[n, k]$ linear code over $\mathbb{F}_q$ and $y \in \mathbb{F}_q^n$. Find a codeword $x \in \mathcal{C}$ where $d(y, x)$ is minimal.*

As we have seen in lemma 2.7 there is a unique solution to the general decoding problem if y can be written as $y = x + e$ with $x \in \mathcal{C}$ and $w(e) \leq \lfloor \frac{1}{2}(d - 1) \rfloor$, but finding x can be very difficult.

Another difficulty in coding theory is the problem of finding weights of a linear code. That is, finding a codeword of weight w for any given natural number w:

Problem 2.12. *Let $\mathcal{C}$ be an $[n, k]$ linear code over $\mathbb{F}_q$ and $w \in \mathbb{N}$; find $x \in \mathcal{C}$ satisfying $d(0, x) = w$.*

An important result for constructing secure cryptosystems is the next lemma.

Lemma 2.13. *The general decoding problem and the problem of finding weights are $\mathcal{NP}$-hard.*

Proof. See [3]. □

We present another problem, based on the equivalence of codes:

Definition 2.14. Two $[n, k]$ codes $\mathcal{C}$ and $\mathcal{C}'$ over $\mathbb{F}_q$ are called *permutation equivalent* if there exists a permutation $\pi \in \mathcal{S}_n$ such that

$$\mathcal{C}' = \pi(\mathcal{C}) = \{(x_{\pi(1)}, \ldots, x_{\pi(n)}) \colon x \in \mathcal{C}\}.$$

Given two generator matrices G and G' the problem is to decide if the codes generated by the matrices are permutation equivalent or not.

In section 3.4.2 we will introduce an algorithm which solves this problem.

2.3 Cyclic Codes

Cyclic codes are the most studied codes of all, since they are easy to encode, and include many important families of codes, for example BCH codes. Furthermore they are building blocks for many other codes, such as the Justesen codes.

First we define cyclic codes in terms of cyclic shifts. We will then study what this means in algebraic terms.

Definition 2.15. The *cyclic shift* $\sigma(c)$ of a word $c = (c_0, c_1, \ldots, c_{n-1}) \in \mathbb{F}_q^n$ is defined by

$$\sigma(c) = (c_{n-1}, c_0, c_1, \ldots, c_{n-2}).$$

A *cyclic code* of length n is a linear code $\mathcal{C}$ in $\mathbb{F}_q^n$, such that

$$\sigma(c) \in \mathcal{C} \quad \text{for all } c \in \mathcal{C}.$$

The subspaces $\{0\}$ and $\mathbb{F}_q^n$ are called the *trivial* cyclic codes.

Example 2.16. *The binary code $\mathcal{C}_3 = \{000, 110, 101, 011\}$ is cyclic.*

Proposition 2.17. *Let G be a generator matrix of a linear code $\mathcal{C}$. Then $\mathcal{C}$ is cyclic if and only if the cyclic shift of every row of G is in $\mathcal{C}$.*

Proof. If $\mathcal{C}$ is cyclic, then the cyclic shift of every row of G is in $\mathcal{C}$, since all the rows of G are codewords.

Conversely, assume that the cyclic shift of every row of G is in $\mathcal{C}$. Let $g_1, \ldots, g_k$ be the rows of G. Let $c \in \mathcal{C}$. Then $c = \sum_{i=1}^{k} x_i g_i$ for some $x_1, \ldots, x_k \in \mathbb{F}_q$. Now σ is a linear transformation of $\mathbb{F}_q^n$. So

$$\sigma(c) = \sum_{i=1}^{k} x_i \sigma(g_i) \in \mathcal{C},$$

since $\mathcal{C}$ is linear and $\sigma(g_i) \in \mathcal{C}$ for all i by assumption. Hence $\mathcal{C}$ is cyclic. □

Proposition 2.18. *The dual of a cyclic code is again cyclic.*

Proof. Let $\mathcal{C}$ be a cyclic code. Then $\sigma(c) \in \mathcal{C}$ for all $c \in \mathcal{C}$, so

$$\sigma^{n-1}(c) = (c_1, \ldots, c_{n-1}, c_0) \in \mathcal{C} \text{ for all } c \in \mathcal{C}.$$

Let $x \in \mathcal{C}^{\perp}$. Then

$$\sigma(x) \cdot c = x_{n-1}c_0 + x_0 c_1 + \cdots + x_{n-2}c_{n-1} = x \cdot \sigma^{n-1}(c) = 0$$

for all $c \in \mathcal{C}$. Hence $\mathcal{C}^{\perp}$ is cyclic. □

We will now show how cyclic codes can be expressed in algebraic terms. The polynomial ring $R_n := \mathbb{F}_q[X]/(X^n - 1)$ will play an important role.

Consider the map $\varphi\colon \mathbb{F}_q^n \to R_n$

$$\varphi(c) = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1},$$

where x^i is the coset of X^i modulo $(X^n - 1)$. The coset of a polynomial $c(X)$ modulo $(X^n - 1)$ is denoted by $c(x)$.

Proposition 2.19. *The map φ is an isomorphism of vector spaces. Ideals in the ring R_n correspond one-to-one to cyclic codes in $\mathbb{F}_q^n$.*

Proof. The map φ is clearly linear.

After division by $X^n - 1$ every polynomial $f(X)$ has a remainder $b(X)$ of degree at most $n-1$. In other words, there exist polynomials $a(X)$ and $b(X)$ such that

$$f(X) = a(X)(X^n - 1) + b(X) \quad \text{and} \quad \deg b(X) < n \quad \text{or} \quad b(X) = 0.$$

So $f(X) \equiv b(X) \mod (X^n - 1)$. Hence the cosets $1, x, \dots, x^{n-1}$ form a basis of R_n. The linear map φ maps the i-th standard basis vector of $\mathbb{F}_q^n$ to the coset x^{i-1} in $\mathbb{F}_q[X]/(X^n - 1)$ for $i = 1, \dots, n$. Hence φ is an isomorphism of vector spaces.

Let ψ denote the inverse map of φ and let I be an ideal in R_n. Then $\mathcal{C} := \psi(I)$ is a linear code, since ψ is a linear map. Let $c \in \mathcal{C}$. Then $c(x) := \varphi(c) \in I$ and I is an ideal; so $xc(x) \in I$. But

$$xc(x) = c_0 x + c_1 x^2 + \cdots + c_{n-2}x^{n-1} + c_{n-1}x^n = c_{n-1} + c_0 x + \cdots + c_{n-2}x^{n-1}$$

since $x^n = 1$. So $\psi(xc(x)) = (c_{n-1}, c_0, \dots, c_{n-2}) \in \mathcal{C}$. Hence $\mathcal{C}$ is cyclic.

A similar proof shows that if $\mathcal{C}$ is a cyclic code in $\mathbb{F}_q^n$, then $I := \varphi(\mathcal{C})$ is an ideal in R_n. □

The codewords of a polynomial code are the elements of an ideal in the ring R_n. As these correspond to cyclic codes in $\mathbb{F}_q^n$, it is sometimes easier to use their inverse images under φ. These are exactly the vectors $c = (c_0, \dots, c_{n-1})$ made up from the coefficients of the polynomials.

A particularly simple kind of ideal is a *principal ideal*, which consists of all multiples of a fixed polynomial $g(x)$, where the scalars are elements of R_n. It will be denoted by

$$\langle g(x) \rangle .$$

The polynomial $g(x)$ is called the *generator polynomial* of the ideal.

In fact every ideal in R_n is a principal ideal; every cyclic code has a generator polynomial. The next theorem proves this and other basic properties of cyclic codes.

Theorem 2.20. *Let $\mathcal{C}$ be a non-zero ideal in R_n, i.e. a cyclic code of length n.*

(a) There is a unique monic[1] polynomial $g(x)$ of minimal degree in $\mathcal{C}$.

(b) $\mathcal{C} = \langle g(x) \rangle$, i.e. $g(x)$ is a generator polynomial of $\mathcal{C}$.

(c) $g(X)$ is a factor of $X^n - 1$.

(d) Any $c(x) \in \mathcal{C}$ can be written uniquely as $c(X) = f(X)g(X)$ in $\mathbb{F}_q[X]$, where $f(X) \in \mathbb{F}_q[X]$ has degree less than $(n-r)$ and $r = \deg g(X)$. The dimension of $\mathcal{C}$ is $(n-r)$. Thus the message $f(X)$ becomes the codeword $f(X)g(X)$.

[1] A polynomial $g(X) = g_0 + g_1 X + \cdots + g_l X^l$ is called *monic* if its leading coefficient g_l is equal to 1.

(e) *If $g(x) = g_0 + g_1 x + \cdots + g_r x^r$, then $\mathcal{C}$ is generated (as a subspace of $\mathbb{F}_q^n$) by the rows of the generator matrix*

$$G = \begin{pmatrix} g_0 & g_1 & g_2 & \cdots & g_r & & 0 \\ & g_0 & g_1 & \cdots & g_{r-1} & g_r & \\ & & & \cdots & \cdots & & \\ 0 & & g_0 & \cdots & \cdots & & g_r \end{pmatrix}$$

$$= \begin{pmatrix} g(x) & & & \\ & xg(x) & & \\ & & \cdots & \\ & & & x^{n-r-1}g(x) \end{pmatrix}.$$

Proof. (a) Suppose $f(x), g(x) \in \mathcal{C}$ are both monic and have minimal degree r. But then $f(x) - g(x) \in \mathcal{C}$ has lower degree, a contradiction unless $f(x) = g(x)$.

(b) Suppose $c(x) \in \mathcal{C}$. Write $c(x) = q(x)g(x) + r(x)$ in R_n, where $\deg r(x) < r$. But $r(x) = c(x) - q(x)g(x) \in \mathcal{C}$ since the code is linear, so $r(x) = 0$. Therefore $c(x) \in \langle g(x) \rangle$.

(c) Write $X^n - 1 = h(X)g(X) + r(X)$ in $\mathbb{F}_q[X]$, where $\deg r(X) < r$. In R_n, this implies $r(x) = -h(x)g(x) \in \mathcal{C}$, a contradiction unless $r(x) = 0$.

(d), (e) From (b), any $c(x) \in \mathcal{C}$, $\deg c(x) < n$, is equal to $q(x)g(x)$ in R_n. Thus

$$\begin{aligned} c(X) &= q(X)g(X) + e(X)(X^n - 1) \text{ in } \mathbb{F}_q[X] \\ &= (q(X) + e(X)h(X))g(X) \text{ in } \mathbb{F}_q[X] \\ &= f(X)g(X) \text{ in } \mathbb{F}_q[X], \end{aligned}$$

where $\deg f(X) \leq n - r - 1$. Thus the code consists of multiples of $g(X)$ by polynomials of degree $\leq n - r - 1$, evaluated in $\mathbb{F}_q[X]$ (not in R_n). There are $n - r$ linearly independent multiples of $g(X)$, namely $g(X), Xg(X), \ldots, X^{n-r-1}g(X)$. The corresponding vectors are the rows of G. Thus the code has dimension $n - r$.

□

The following is an example of a cyclic code including its generator polynomial.

Example 2.21. *The parity check matrix of a binary Hamming code $\mathcal{H}_m$ of length $n = 2^m - 1$ has as columns all $2^m - 1$ distinct non-zero m-tupels.*

Thus, for $\mathcal{H}_3$,

$$H = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Therefore the generator polynomial is $g(x) = 1 + x + x^3$ and we get as a generator matrix

$$G = \begin{pmatrix} 1 & 1 & 0 & 1 & & & \\ & 1 & 1 & 0 & 1 & & \\ & & 1 & 1 & 0 & 1 & \\ & & & 1 & 1 & 0 & 1 \end{pmatrix}.$$

Using a matrix over $\mathbb{F}_{q^m}$ to define a code over $\mathbb{F}_q$ A situation often arising in the context of polynomial codes is that we want to define a code over $\mathbb{F}_q$ using a matrix over $\mathbb{F}_{q^m}$.

First, suppose the code is to be defined by a parity check matrix H over $\mathbb{F}_{q^m}$. More precisely, let $H = (H_{ij})$, where $H_{ij} \in \mathbb{F}_{q^m}$, be an $r \times n$ matrix of rank r over $\mathbb{F}_{q^m}$. Then let $\mathcal{C}_H$ be the code over $\mathbb{F}_q$ consisting of all vectors $a = (a_1, \ldots, a_n)$, $a_i \in \mathbb{F}_q$, such that $Ha^T = 0$.

Another way of getting $\mathcal{C}_H$ is as follows. Pick a basis $\alpha_1, \ldots, \alpha_m$ for $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$, and write

$$H_{ij} = \sum_{i=1}^{m} H_{ijl}\alpha_l, \quad H_{ijl} \in \mathbb{F}_q.$$

Define $\bar{H}$ to be the $rm \times n$ matrix obtained from H by replacing each entry H_{ij} by the corresponding column vector $(H_{ij1}, \ldots, H_{ijm})^T$ from $\mathbb{F}_q$. Thus

$$\bar{H} = \begin{pmatrix} H_{111} & H_{121} & \ldots & H_{1n1} \\ H_{112} & H_{122} & \ldots & H_{1n2} \\ \vdots & \vdots & \ddots & \vdots \\ H_{11m} & H_{12m} & \ldots & H_{1nm} \\ \vdots & \vdots & \ddots & \vdots \\ H_{r1m} & H_{r2m} & \ldots & H_{rnm} \end{pmatrix}.$$

Then

$$\begin{aligned} a \in \mathcal{C}_H &\Leftrightarrow \sum_{j=1}^{n} H_{ij}a_j = 0 \quad \text{for } i = 1, \ldots, r \\ &\Leftrightarrow \sum_{j=1}^{n} H_{ijl}a_j = 0 \quad \text{for } i = 1, \ldots, r; \ \ l = 1, \ldots, m \\ &\Leftrightarrow \bar{H}a^T = 0. \end{aligned}$$

Thus either H or $\bar{H}$ can be used to define $\mathcal{C}_H$. The rank of $\bar{H}$ over $\mathbb{F}_q$ is at most rm, so $\mathcal{C}_H$ is an $[n, k \geq n - rm]$ code, assuming $rm \leq n$.

Of course we could also consider the code $\mathcal{C}_H^{\#}$ over $\mathbb{F}_{q^m}$ consisting of all vectors $b = (b_1, \ldots, b_n)$, $b_i \in \mathbb{F}_{q^m}$, such that $Hb^T = 0$. Then $\mathcal{C}_H^{\#}$ is an $[n, n - r]$ code over F_{q^m}. Since $\mathbb{F}_q \subset \mathbb{F}_{q^m}$, every codeword in $\mathcal{C}_H$ is in $\mathcal{C}_H^{\#}$. In fact, $\mathcal{C}_H$ consists of exactly those codewords of $\mathcal{C}_H^{\#}$ which have components from $\mathbb{F}_q$. We will denote this by writing

$$\mathcal{C}_H = \mathcal{C}_H^{\#}|_{\mathbb{F}_q}$$

and call $\mathcal{C}_H$ a *subfield subcode* of $\mathcal{C}_H^{\#}$.

In the next section, we take a look at the McEliece public-key cryptosystem. As we want to investigate the use of certain codes for this cryptosystem, we will describe the system in detail and show some of its properties.

3 McEliece Public-Key Encryption

The McEliece public-key encryption scheme is based on error-correcting codes. The idea behind this scheme is to first select a particular (linear) code for which an efficient decoding algorithm is known, and then to use a trapdoor function to disguise the code as a general linear code. Since the problem of decoding an arbitrary linear code is NP-hard, a description of the original code can serve as the private key, while a description of the transformed code serves as the public key.

The McEliece encryption scheme (when used with Goppa codes, as originally proposed by McEliece in 1978) has resisted cryptoanalysis to date. It is also notable as being the first public-key encryption scheme to use randomization in the encryption process. Although very efficient, the McEliece encryption scheme has received little attention in practice because of the very large public keys.

3.1 The McEliece Cryptosystem

Let $\mathcal{C}$ be an $[n, k]$ linear code with a fast decoding algorithm that can correct up to t errors. Let G be a generator matrix for $\mathcal{C}$. To create the disguise, let S be a random $k \times k$ invertible matrix (the *scrambler*) and let P be a random $n \times n$ permutation matrix. The matrix

$$\widehat{G} = SGP$$

is made public while S, G and P form the private key.

Encryption: Represent the message as a string m of length k, choose a random error vector e of weight at most t and compute the ciphertext $c = m\widehat{G} + e$.

Decryption: To recover the plaintext m from c, we compute $\widehat{c} = cP^{-1}$, use the decoding algorithm for the code generated by G to decode $\widehat{c}$ to $\widehat{m}$ and compute $m = \widehat{m}S^{-1}$.

Proof that decryption works. Since

$$\widehat{c} = cP^{-1} = (m\widehat{G} + z)P^{-1} = (mSGP + z)P^{-1} = (mS)G + zP^{-1}$$

and zP^{-1} has weight at most t, the decoding algorithm for the code generated by G corrects $\widehat{c}$ to $\widehat{m} = mS$. Finally, $\widehat{m}S^{-1} = m$ and, hence, decryption works. □

3.2 Practicality of the McEliece Scheme

As pointed out by Rao and Nam [58], the McEliece scheme requires rather large block length. They suggested $n = 1024$, but today this is not enough anymore, so $n = 2048$ should be chosen. Therefore this scheme produces too much computational overhead for encryption and decryption for most practical applications.

3.3 Extensions

There are some extensions to this scheme. The first one was developed by F. Jorissen [35]. The idea was to add only $t' < t$ errors, such that $t - t'$ additional errors can be corrected. This implies that the security level degrades, but it can be very useful if the message is sent through a noisy channel.

A second idea consists of improving the code rate by transferring some data through the pattern of the error bits [13, 53]. This has no effect on the security if the data in the concealed channel is perfectly random, otherwise the attacker can take an important advantage of it.

3.4 Cryptanalytic Attacks

In this section, we review currently known attacks to the McEliece Cryptosystem (see McEliece [49], Kobara and Imai [40, 32], Adams and Meijer [1], Rao and Nam [58], Park [53], Gibbon [26], Korzhik and Turkin [41], Tilburg [78], Lee and Brickell [42], Simmons [17], Beth et al. [6], Loidreau and Sendrier [45], Berson [5], Sidelnikov and Shestakov [71], and Engelbert, Overbeck and Schmidt [21] for these and other attacks). The classification follows Kobara and Imai [40].

While no efficient algorithm for decomposing G' into (S, G, P) has been discovered yet [50], a structural attack has been discovered in [45]. This attack reveals part of the structure of a weak G' which is generated from a binary Goppa polynomial. However, this attack can be avoided simply by not using such weak public keys. This implies G should not be a BCH code since this would be equivalent to a Goppa code whose Goppa polynomial is $1 + x^{2t}$, i.e. binary.

The next case we have to consider is that an equivalent Goppa code of G' (which is not nevessarily G), whose decoding algorithm is known, happens to be found. This probability is estimated in [1] and [26], and then shown to be negligibly small.

All other known attacks are used to decrypt ciphertexts without breaking public-keys. We classify them into the two categories critical and non-critical attacks. Non-critical attacks are those that depend strongly on the parameters and can thus be rendered infeasible just by enlarging the parameter sizes. Critical attacks are much faster than the non-critical ones and are therefore feasible for realistic parameter sizes. They can only be avoided by using suitable conversions (see Kobara and Imai [40]) and by avoiding codes having a structural weakness.

Interestingly, all the critical attacks exploit structural weaknesses of the codes used, or require additional information, such as partial knowledge on the target plaintexts. Without this additional information, no efficient algorithm is known to decrypt an arbitrarily given ciphertext of the McEliece PKC.

3.4.1 Non-Critical Attacks

The following two attacks can be rendered infeasible simply by enlarging the parameter size. Kobara and Imai suggested in [40] to apply Loidreau's modification from [44] to further increase the work factor. In [32] though, they show that this modification, while increasing the work factor against ever known chosen-plaintext attacks (CPA), is vulnerable against a new CPA they developed. Thus, it is rather harmful to apply this modification to the McEliece cryptosystem.

Generalized Information-Set-Decoding Attack Of the known general attacks (i.e., not against specific codes etc.) this seems to have the lowest complexity. One tries to recover the k information symbols as follows: The first step is to pick k of the n coordinates randomly in the hope that none of the k are in error. We then try to recover the message by solving the $k \times k$ linear system (binary or over $\mathbb{F}_q$).

Let G'_k, c_k and z_k denote the k columns picked from G', c and z, respectively. They have the following relationship

$$c_k = mG'_k + z_k.$$

If $z_k = 0$ and G'_k is non-singular, m can be recovered by

$$m = c_k {G'}_k^{-1}.$$

The computation cost of the this version (called the *original information-set-decoding attack*) is $T(k) \cdot P^{-1}_{n,k,t}$, where

$$P_{n,k,t} = \prod_{i=0}^{k-1} \left(1 - \frac{t}{n-i}\right).$$

The quantity $T(k)$ in the average work factor is the number of operations required to solve a $k \times k$ linear system over $\mathbb{F}_q$. As mentioned in [49], solving a $k \times k$ binary system takes about k^3 operations. Over $\mathbb{F}_q$, it would require at least $(k \cdot \log_2 q)^3$ operations.

Even if $z_k \neq 0$, m can be recovered by guessing z_k among small Hamming weights [42, 11] (this is called the *generalized information-set-decoding (GISD) attack*). One iteration of the algorithm is as follows:

1. Permute the columns of the generator matrix randomly.
2. Apply gaussian elimination on the rows of the matrix to obtain the form $G = (I_k|A)$, with the corresponding permuted cipher text $c = (c_1 + e_1|c_2 + e_2)$.
3. Guess that the error e_1 is of weight at most p and check whether the error $e = (e_1|e_2)$ is of weight t.

The probability π that a permutation of the columns leads to a favorable configuration is

$$\pi(p, n, k, t) = \sum_{i=0}^{p} \frac{\binom{n-t}{k-i}\binom{t}{i}}{\binom{n}{k}}.$$

For each iteration, an estimate for the number of operations is

1. $\frac{k^2 n}{2}$ for the gaussian elimination.
2. About $k/2 + \sum\limits_{i=1}^{p} \binom{k}{i} i$ additions on the $(n-k)$-bit words of A.

Thus, an estimate of the work factor of this algorithm is

$$W(p, n, k, t) = \frac{\frac{k^2 n}{2} + (n-k)\left[k/2 + \sum_{i=1}^{p} \binom{k}{i} i\right]}{\pi(p, n, k, t)}.$$

The generalized version of this algorithm and its computational cost was investigated by F. Chabaud [11], amongst others. It is slightly faster than the original version (where z_k is assumed to be 0), but it is still infeasible for appropriate parameters (see [40]).

Finding-Low-Weight-Codeword Attack This attack uses an algorithm which finds a low-weight codeword among codewords generated by an arbitrary generator matrix using a database obtained by pre-computation [73, 10]. Since the minimum-weight codeword of the following $(k+1) \times n$ generator matrix

$$\begin{bmatrix} G' \\ c \end{bmatrix}$$

is the error vector z of c where $c = mG' + z$, this algorithm can be used to recover m from a given ciphertext c.

The precise computational cost of this attack is evaluated in [10]. It is shown to be infeasible to invert c for appropriate parameters, e.g. $n \geq 2048$ and optimized k and t, even though using the original parameters $(n, k, t) = (1024, 524, 50)$ suggested in [49], it is feasible with the work factor of $2^{64.2}$.

3.4.2 Critical Attacks

The following attacks cannot be avoided by enlarging the parameter size; they target specific structural weaknesses or need additional information.

Known-Partial-Plaintext Attack Having partial knowledge on the target plaintext drastically reduces the computational cost of the attacks to the McEliece PKC [10, 39].

For example, let m_l and m_r denote the left k_l bits and the remaining k_r bits in the target plaintext m, i.e. $k = k_l + k_r$ and $m = (m_l | m_r)$. Suppose that an adversary knows m_r. Then the difficulty of recovering the unknown plaintext m_l in the McEliece PKC with parameters (n, k) is equivalent to that of recovering the full plaintext in the McEliece PKC with parameters (n, k_l), since

$$\begin{aligned} c &= mG' + z \\ c &= m_l G'_l + m_r G'_r + z \\ c + m_r G'_r &= m_l G'_l + z \\ c' &= m_l G'_l + z \end{aligned}$$

where G'_l and G'_r are the upper k_l rows and the remaining lower k_r rows in G', respectively.

Message-Resend Attack Suppose now that, through some accident, or as a result of action in the part of the cryptanalyst, both

$$c_1 = mG' + e_1$$

and

$$c_2 = mG' + e_2,$$

$e_1 \neq e_2$, are sent. This is called a *message-resend* condition. I this case it is easy for the cryptanalyst to recover m from the system of c_i. We will only examine the case where $i = 2$. The attack is even easier for $i > 2$.

Notice that $c_1 + c_2 = e_1 + e_2 \pmod 2$.

A message-resend condition can easily be detected by observing the Hamming weight of the sum of any two cryptograms. When the underlying messages are different, the expected weight of the sum is about 512 (for the original parameters of McEliece; in general, the expected weight is k). When the underlying messages are identical, the weight of the sum cannot exceed 100 (or, in general, $2t$). Heiman [29] showed that a message-resend condition can be detected; we will show how to exploit it. In the following we will use the original parameters $(n, k, t) = (1024, 524, 50)$. The results for other parameters are essentially the same.

First we compute two sets from $(c_1 + c_2)$. The set L_0 will consist of the locations where $(c_1 + c_2)$ contains zeros. The set L_1 will consist of the locations where $(c_1 + c_2)$ contains ones.

$$L_0 = \{l \in \{1..n\} \colon c_1(l) + c_2(l) = e_1(l) + e_2(l) = 0\}$$
$$L_1 = \{l \in \{1..n\} \colon c_1(l) + c_2(l) = e_1(l) + e_2(l) = 1\}$$

We aim to take advantage of the fact that

- $l \in L_0 \Rightarrow$ most probably neither $c_1(l)$ nor $c_2(l)$ is garbled by an error, while
- $l \in L_1 \Rightarrow$ certainly precisely one of $c_1(l)$ or $c_2(l)$ is garbled by an error.

Assuming the error vectors e_1 and e_2 are chosen independently, then for any $l \in \{1..n\}$ the probability that both error vectors are 1 at location l is

$$P(e_1(l) = e_2(l) = 1) = \left(\frac{50}{1024}\right)^2 \approx 0.0024.$$

In other words, most $l \in L_0$ signify $e_1(l) = e_2(l) = 0$. Thus, the cryptanalyst should try to guess the 524 ungarbled columns from those indexed by L_0.

How good is this strategy? Let p_i be the probability that precisely i coordinates are simultaneously garbled by e_1 and e_2. Then

$$p_i = P\left(|\{l : e_1(l) = 1\} \cap \{l : e_2(l) = 1\}| = i\right) = \frac{\binom{50}{i}\binom{974}{50-i}}{\binom{1024}{50}},$$

since, say, e_2 must choose i error locations from those 50 garbled by e_1 and the remaining $50 - i$ from those unchanged by e_1. Therefore, the expected cardinality of L_1 is

$$E(|L_1|) = \sum_{i=0}^{50} (100 - 2i)p_i \approx 95.1,$$

since every l for which $e_1(l) = e_2(l) = 1$ reduces $|L_1|$ by two.

For example, suppose $|L_1| = 94$. Then $|L_0| = 930$, of which only 3 are garbled. We see that the probability of guessing 524 ungarbled columns from those indexed by L_0 is

$$\frac{\binom{927}{524}}{\binom{930}{524}} \approx 0.0828,$$

so the cryptanalyst can expect to succeed in this case with only 12 guesses.

These results are better by a factor of 10^{15} than guessing k ungarbled columns without message-resend condition.

Related-Message Attack We will now generalize the message-resend attack. Suppose that there are two cryptograms

$$c_1 = m_1G' + e_1$$

and

$$c_2 = m_2G' + e_2,$$

where $e_1 \neq e_2$. The messages m_1 and m_2 can differ, but we assume the cryptanalyst knows a linear relation, for example m_1+m_2, between them. This is called a *related-message* condition. In this case the cryptanalyst may recover the m_i from the set of c_i by doing one encoding and by then using the previous attack method:

Combining the two cryptograms we get

$$c_1 + c_2 = m_1G' + m_2G' + e_1 + e_2.$$

Notice that $m_1G'+m_2G' = (m_1+m_2)G'$, a value the cryptanalyst may calculate in a related-message condition from the known relationship and the public key.

He then solves

$$c_1 + c_2 + (m_1 + m_2)G' = e_1 + e_2$$

and proceeds with the message-resend attack, using $(c_1+c_2+(m_1+m_2)G')$ in place of (c_1+c_2).

Remark. The message-resend attack is the special case of the related-message attack where $m_1 + m_2 = 0$.

Appart from these general attacks there are some attacks targeting McEliece Cryptosystems using specific codes. Of these we will show two important attacks, one against Generalized Reed-Solomon (GRS) codes and one against Goppa codes generated by a binary generator polynomial.

McEliece using GRS-Codes Though the following attack originally targeted the Niederreiter cryptosystem, it can be adapted to the McEliece cryptosystem, as Li, Deng and Wang [43] show the equivalence of both systems.

In 1992 Sidelnikov and Shestakov proposed an attack on Niederreiter's cryptosystem using Generalized Reed-Solomon (GRS) codes [71] which aims to recover an alternative private key from the public key. They take advantage of the fact that the check matrix of a GRS code is of the form

$$H = \begin{pmatrix} z_1 a_1^0 & z_1 a_1^1 & \cdots & z_1 a_1^s \\ z_2 a_2^0 & z_2 a_2^1 & \cdots & z_2 a_2^s \\ \vdots & \vdots & \ddots & \vdots \\ z_n a_n^0 & z_n a_n^1 & \cdots & z_n a_n^s \end{pmatrix} \in \mathbb{F}_q^{n \times (s+1)}.$$

Sidelnikov and Shestakov concluded that each entry of the public key matrix H' can be expressed by a polynomial in a_i. From this observation one can derive a system of polynomial equations whose solution yields the private key.

Using this method, it is possible to decypher the message in polynomial time.

McEliece using Goppa codes with binary generator polynomial Pierre Loidreau and Nicolas Sendrier showed in 2001 [45] that it is possible to know whether the secret Goppa code of an instance of the McEliece cryptosystem was chosen with a binary generator polynomial. Furthermore they presented an attack which, whenever such a weak key is used, can be completed with a large, but feasible amount of computations.

First they showed that the automorphism group of a Goppa code with binary generator polynomial is generated by the Frobenius field automorphism. This can be used to detect if a weak key has been used. However, an exhaustive search over the space of all such codes remains too costly.

Loidreau and Sendrier found another way to reduce the search space of a brute force attack. Their idea was to take advantage of the *Support Splitting Algorithm* (SSA)[2] presented in [66]. SSA can be used to decide whether two codes are permutation equivalent. They used this ability by performing an exhaustive search on the space of binary irreducible polynomials of degree t and checking the codes they generate for equivalence to the given code with SSA.

3.4.3 Conclusions

The parameters (n, k, t) of the McEliece cryptosystem must be chosen large enough to render the (generalized) information-set-decoding attack infeasible. A suitable size for n seems to be ≥ 2048 (if using Goppa codes). The size of k and t should be optimized to yield high security while keeping the key sizes small. Here $[n, k, t] = [2048, 1608, 81]$ (small public key) and $[k, t] = [2048, 1278, 141]$ (maximum security) seem to be good values.

Neither GRS codes, nor Goppa codes generated by a binary generator polynomial (nor BCH codes, for the same reason) should be used, as there are structural attacks against those resulting in much smaller workloads.

Gabidulin [25] proposed a modification to the cryptosystem to avoid the attack against GRS codes which remains unbroken, but the result is too recent to have been evaluated.

[2] for more information on the SSA, see [21].

4 Algebraic-Geometric Codes

In this section we will give an introduction to algebraic geometry and show how to use it to construct codes. A good overview of algebraic-geometric codes can be found in [75] and [74].

4.1 Introduction

What are *Algebraic-Geometric (AG) Codes* and why do we want to use them?

In the early 1980s the russian mathematician V. D. Goppa [27] had the idea of associating to a curve χ defined over $\mathbb{F}_q$ a code $\mathcal{C}$.

One of the main features of Goppa's construction is that the minimum distance d of $\mathcal{C}$ is bounded from below, whereas in general there is no lower bound available on the minimum distance of a code.

Another reason for the interest in algebraic-geometric codes is the fact that those codes can be used to give an asymptotically good sequence of codes with parameters better than the so-called Varshamov-Gilbert bound in a certain range of the rate and for large enough alphabets. The construction can be found in [8].

As a motivation for the construction of AG codes, we first consider *Reed Solomon Codes* over $\mathbb{F}_q$. This important class of codes has been well-known in coding theory for a long time. AG codes are a natural generalization of Reed Solomon codes.

Let q be a prime power, n and k integers such that $1 \leq k \leq n \leq q$. Let $\mathbb{F}_q[X]$ be the ring of polynomials in one variable with coefficients in $\mathbb{F}_q$. Now set

$$\mathcal{L}_k := \{f \in \mathbb{F}_q[X] \colon \deg f \leq k-1\}.$$

For n distinct elements $P_1, \ldots, P_n$ of $\mathbb{F}_q$, consider the following $\mathbb{F}_q$-linear (evaluation) map:

$$\begin{aligned} e : \mathcal{L}_k &\to \mathbb{F}_q^n \\ f &\mapsto (f(P_1), \ldots, f(P_n)) \end{aligned}$$

We know that e is injective since a non-zero polynomial in $\mathcal{L}_k$ can have at most $(k-1)$ zeros. Then the (linear) code $\mathcal{C} = e(\mathcal{L}_k)$ has dimension k. The code $\mathcal{C}$ is called a Reed-Solomon code.

4.2 Algebraic Curves

Before we can generalize Reed-Solomon codes we have to understand some basic concepts from algebraic geometry. First we need to define algebraic curves.

Definition 4.1. An *affine space* is a set with a transitive vector space action (an action on the additive group of a vector space having only one orbit). Alternatively, an affine space is a set S, together with a vector space V and a map

$$\Theta : S^2 \to V, (a,b) \mapsto \Theta(a,b)$$

such that

1. for every $b \in S$ the map
$$\Theta_b : S \to V, a \mapsto \Theta(a, b)$$
is a bijection, and
2. for every $a, b, c \in S$ we have
$$\Theta(a, b) + \Theta(b, c) = \Theta(a, c)$$

If n is the dimension of the vector space V, then S is also called an *affine n-space.*

As there is no distinguished element in S we can informally say that "an affine space is a vector space that has forgotten its origin" (John Baez).

Definition 4.2. Let $\mathbb{F}$ be a field and let $\mathbb{F}^n$ be affine n-space over $\mathbb{F}$. The polynomials f in $\mathbb{F}[X_1, \dots, X_n]$ form a ring called the *coordinate ring* of $\mathbb{F}^n$.

By considering the set of common zeros of a set of polynomials, each subset T of the coordinate ring of $\mathbb{F}^n$ determines a subset $Z(T)$ of this affine space (and vice versa).

Definition 4.3. A subset V of $\mathbb{F}^n$ is called an *affine algebraic set* if $V = Z(T)$ for some subset T of the coordinate ring. A nonempty affine algebraic set V is called *irreducible* if it cannot be written as the union of two proper affine algebraic subsets. An irreducible affine algebraic set is called an *affine variety.*

To illustrate these concepts we give an example.

Example 4.4. *The affine algebraic set $T := \{(x, y) \in \mathbb{R}^2 \colon xy = 0\}$ is not irreducible, as it can be written as the union of the two coordinate axes $T = \{(x, 0) \colon x \in \mathbb{R}\} \cup \{(0, y) \colon y \in \mathbb{R}\}$, which are affine algebraic sets as well.*

When we study polynomials on a subset of affine space only, we do not want to distinguish between functions being identical on that subset.

Definition 4.5. Let $I(S)$ be the ideal of all functions vanishing on an affine algebraic variety S. The quotient of the polynomial ring by this ideal is the *coordinate ring of the affine algebraic variety S.*

The importance of affine algebraic varieties can be seen in the following lemma:

Lemma 4.6. *Let V be an affine algebraic variety, then $I(V)$ is a prime ideal in $\mathbb{F}[X_1, \dots, X_n]$.*

Proof. Write $\mathfrak{a} = I(V)$ and suppose that $\mathfrak{a}$ is not prime. Then, if $V \neq \emptyset$, $\mathfrak{a}$ is a proper subset of $F[X_1, \dots, X_n]$ and there exist $f_1, f_2 \notin \mathfrak{a}$ but $f_1 f_2 \in \mathfrak{a}$. Hence there exist $p_1, p_2 \in V$ such that $f_i(p_i) \neq 0$. Put $V_i = Z(\mathfrak{a} + (f_i))$, $i = 1, 2$, then V_i is affine algebraic and a proper subset of V. Moreover
$$V_1 \cup V_2 = Z((\mathfrak{a} + (f_1))(\mathfrak{a} + (f_2))) = Z(\mathfrak{a}^2 + (f_1 f_2)) = V,$$
thus V is reducible. □

Definition 4.7. The *dimension* of an algebraic variety V is defined as the *height* of the corresponding prime ideal $P = I(V)$, that is, the maximal length of an ascending chain of prime ideals

$$P_0 \subsetneq P_1 \subsetneq \ldots \subsetneq P_n = P.$$

Definition 4.8. An *algebraic curve* χ is an affine variety of dimension equal to 1.

Remark. In many cases our algebraic curves will be *plane affine algebraic curves*, i.e. curves defined by an equation

$$\chi = \{(X, Y) \in \mathbb{F}^2 \colon F(X, Y) = 0\},$$

where $F \in \mathbb{F}[X, Y]$ is a non-constant polynomial over $\mathbb{F}$.

To simplify matters, the rest of this chapters deals only with plane affine algebraic curves.

Notation 4.9. *If χ is a plane affine algebraic curve, then the coordinate ring of χ is written $\mathbb{F}[\chi]$.*

Definition 4.10. Let χ be an algebraic curve over $\mathbb{F}_q$. Then the points on χ all of whose coordinates lie in $\mathbb{F}_q$ are called *rational points.*

The reason for using algebraic curves is that we can introduce additional structure to the ring of polynomials.

Definition 4.11. Let χ be an algebraic curve, then $I(\chi)$ is a prime ideal. Therefore, the coordinate ring is an integral domain and thus has a quotient field $\mathbb{F}(\chi)$ called the *function field of χ.*

As the functions $f \in \mathbb{F}(\chi)$ can be written as $f = g/h$ with $g, h \in \mathbb{F}[\chi]$, the value of f at a (rational) point P is $g(P)/h(P)$, if $h(P) \neq 0$, or undefined.

The set of all functions $f \in \mathbb{F}(\chi)$ that are defined at a given point P form a ring $\mathcal{O}_{\chi,P}$. Abusing the notation we sometimes write $\mathcal{O}_P$ if it is clear which curve χ is meant.

Evaluating at P gives us a surjective ring homomorphism $\mathcal{O}_{\chi,P} \to \mathbb{F}$; its kernel is the maximal ideal $\mathfrak{m}_P = \{f \in \mathcal{O}_{\chi,P} \colon f(P) = 0\}$. This is the only maximal ideal, since every element f in $\mathcal{O}_{\chi,P} \backslash \mathfrak{m}_P$ can be written as $f = g/h$ with both g and h being non-zero. Therefore f is invertible. Hence $\mathcal{O}_{\chi,P}$ is a *local ring*[3].

Theorem 4.12. *If the curve χ is* smooth*, i.e. at least one partial derivative at P is non-zero, then $\mathcal{O}_{\chi,P}$ is a local principal ideal domain, also called* discrete valuation ring.

Thus there exists an element $t \in \mathcal{O}_{\chi,P}$ such that every element $f \in \mathbb{F}(\chi)^\times$ can be uniquely written as $f = ut^n$ with $u \in \mathcal{O}_{\chi,P}^\times$ and $n \in \mathbb{Z}$.

For the proof we are going to need *Nakayama's Lemma*:

Lemma 4.13. *Let R be a local noetherian ring with maximal ideal $\mathfrak{m}$.*

(1) If M is a finitely generated R-module with $M = \mathfrak{m} \cdot M$, then $M = 0$.

[3] a local ring R is a commutative ring that contains a single maximal ideal $\mathfrak{m}$.
One property of a local ring R is that the subset $R \backslash \mathfrak{m}$ is precisely the set of ring units.

(2) Let $N \subset M$ be two R-modules with M/N finitely generated, then

$$M = N + \mathfrak{m} \cdot M \quad \Rightarrow \quad M = N.$$

Proof of Theorem 4.12. The proof consists of two steps:

(1) χ is smooth in $P \Rightarrow \dim \mathfrak{m}_P/\mathfrak{m}_P^2 = 1$.

(2) $\mathcal{O}_{\chi,P}$ is local and noetherian with maximal ideal $\mathfrak{m}$ and $\mathfrak{m}_P/\mathfrak{m}_P^2$ is a 1-dimensional $\mathbb{F}(\chi)/\mathfrak{m}$-vector space implies that $\mathcal{O}_{\chi,P}$ is a discrete valuation ring and that there is an element $t \in \mathcal{O}_{\chi,P}$ with the above property.

In detail:

(1) Let χ be a plane affine algebraic curve defined by a polynomial F such that χ is smooth at P. After a transformation of coordinates we can assume that $P = (0,0)$. Hence F has the form $F(X,Y) = aX + bY +$terms of higher order. As the partial derivatives at P are a and b, respectively, we have $a \neq 0$ or $b \neq 0$. Without loss of generality let $b \neq 0$. Then for every point $(x,y) \in \mathcal{C}$ we have $y \equiv -b^{-1}ax \mod \mathfrak{m}_P^2$. Hence $\mathfrak{m}_P/\mathfrak{m}_P^2$ is generated by $x \mod \mathfrak{m}_P^2$. Nakayama's Lemma states that if $\mathfrak{m}_P = \mathfrak{m}_P^2$ then $\mathfrak{m}_P = 0$. As $\mathfrak{m}_P \neq 0$ we get $\dim \mathfrak{m}_P/\mathfrak{m}_P^2 = 1$.

(2) According to the last condition there is a $t \in \mathfrak{m}$ such that $\mathfrak{m}_P/\mathfrak{m}_P^2 = (\mathbb{F}(\chi)/\mathfrak{m}) \cdot \bar{t}$. Thus, $\mathfrak{m}_P = \mathbb{F}(\chi) \cdot t + \mathfrak{m}_P^2$. It follows from Nakayama's Lemma that $\mathfrak{m}_P = \mathbb{F}(\chi) \cdot t$.

Let $f \in \mathbb{F}(\chi)^\times$. First assume $f \in \mathcal{O}_{\chi,P}$. Set $f_0 = f$ and construct a sequence $(f_n) \in \mathcal{O}_{\chi,P}$. If $f_n \in \mathcal{O}_{\chi,P}^\times$ for some n then set $u = f_n$ and stop. Otherwise $f_n \in \mathfrak{m}$, hence we can write f_n as $f_n = f_{n+1}t$. If this sequence stops, we get $f = ut^n$ with $n \in \mathbb{N}$. If the sequence did not stop we would have a properly increasing chain of ideals $\mathcal{O}_{\chi,P} \cdot f_0 \subsetneq \mathcal{O}_{\chi,P} \cdot f_1 \subsetneq \dots$ (because $f_{n+1} \in \mathcal{O}_{\chi,P} \cdot f_n = \mathcal{O}_{\chi,P} \cdot f_{n+1} \cdot t$ would imply $t \in \mathcal{O}_{\chi,P}^\times$). This is not possible as $\mathcal{O}_{\chi,P}$ is noetherian.

If $f \notin \mathcal{O}_{\chi,P}$ then f can be written as $f = g/h$ with $g, h \in \mathbb{F}[\chi]$ and $g(P) \neq 0$, $h(P) = 0$. Then $f^{-1} = h/g \in \mathcal{O}_{\chi,P}$. Therefore f^{-1} can be written as $f^{-1} = ut^n$, $n \in \mathbb{N}$ and thus $f = u^{-1}t^{-n}$.

Finally, we want to show that $\mathcal{O}_{\chi,P}$ is a principal ideal domain. Let $0 \neq I \subset \mathcal{O}_{\chi,P}$ be an ideal. Set $n = \min\{v \in \mathbb{N} \colon t^v \in I\}$. Now $\mathcal{O}_{\chi,P} \cdot t^n \subseteq I$ as $t^n \in I$ by definition. The other inclusion follows from $ut^n \in I \Rightarrow t^n \in I$. □

As the exponent n in the representation $f = ut^n$ does not depend on the choice of t, it defines a surjective function $v_P : \mathbb{F}(\chi)^\times \to \mathbb{Z}, f \mapsto n$, where $f = ut^n$ with $u \in \mathcal{O}_{\chi,P}^\times$. This function has the following properties:

$$v_P(fg) = v_P(f) + v_P(g), \quad v_P(f+g) \geq \min\{v_P(f), v_P(g)\}, \quad v_P|_{\mathbb{F}^\times} = 0$$

(we set $v_P(0) = \infty$ and $\infty + n = \infty$ for all $n \in \mathbb{Z}$).

The function v_P can be interpreted as the vanishing order of f at P, or $-v_P$ as the order of the pole of f at P, if $v_P(f) < 0$.

We call this function *discrete valuation of* $\mathbb{F}(\chi)/\mathbb{F}$.

Example 4.14. *As an example, we consider the plane curve χ defined by $Y = 0$, i.e. the X-axis. The function field is $\mathbb{F}(\chi)$ and the (affine) points are $(a, 0)$ with $a \in \mathbb{F}$. The associated* *discrete valuation ring is*

$$\mathcal{O}_a = \{g/h\colon g,h \in \mathbb{F}[\chi], h(a) \neq 0\}.$$

But there is one more discrete valuation ring in $\mathbb{F}(\chi)$*, namely*

$$\mathcal{O}_\infty = \{g/h\colon g,h \in \mathbb{F}[\chi], h \neq 0, \deg h \geq \deg g\}$$

with associated valuation $v_\infty(g/h) = \deg h - \deg g$*. The affine line is missing the point* ∞*. To get this point if we would have to use the projective line.*

4.3 Divisors

We need a way to describe sets of functions that may have poles of given order only at specific points. For that purpose we introduce the group of divisors on χ.

Definition 4.15. The *group of divisors on* χ, $\operatorname{Div}(\chi)$, is the free abelian group over the points of χ. Hence a divisor has the form $D = \sum_P n_P \cdot P$, where P ranges over the points of χ and $n_P \in \mathbb{Z}$ with $n_P = 0$ for all but finitely many P. We write $n_P = v_P(D)$.

Definition 4.16. The *support* of a divisor D is the set of points with non-zero coefficient, i.e. $\operatorname{supp} D = \{P \in \chi\colon n_P \neq 0\}$.

Definition 4.17. The *degree* of D is defined as

$$\deg D = \sum_P n_P \in \mathbb{Z}.$$

To every function we assign a divisor

$$\operatorname{div}(f) = \sum_P v_P(f) \cdot P.$$

So, in a sense, the divisor of f is a bookkeeping device that tells us where the zeros and poles of f are and what their multiplicities and orders are.

On the group of divisors we define a partial order. For two divisors D, D' we say that $D \geq D'$ if the inequality holds coefficientwise, i.e. $n_P \geq n'_P$ for all points $P \in \chi$.

The set

$$L(D) = \{f \in \mathbb{F}(\chi)\colon \operatorname{div}(f) + D \geq 0\} \tag{1}$$

is a finite dimensional vector space consisting of those functions having a pole order not greater than $v_P(D)$ at every point $P \in \chi$.

Some properties of the code we will derive from this construction depend on the size of this vector space. The *Theorem of Riemann-Roch* gives an answer to this: It states that there is a natural number $g = g(\chi)$ and a divisor $\kappa \in \operatorname{Div}(\chi)$, such that for all divisors D

$$\dim L(D) = \deg D - g + 1 + \dim L(\kappa - D) \tag{2}$$

holds.

From (1) we get $L(D) = 0$ if $\deg D < 0$ and $L(0) = \mathbb{F}$. Working with $D = 0$ in the Theorem of Riemann-Roch, (2) shows that $\dim L(\kappa) = g$ and $D = \kappa$ implies $\deg \kappa = 2g - 2$. Hence we get as a corollary

$$\dim L(D) \geq \deg D - g + 1$$

with equality if $\deg D \geq 2g + 1$. Hence g is uniquely defined.

Definition 4.18. The (unique) natural number g that satisfies the Theorem of Riemann-Roch (2) is called the *genus* of the curve χ.

We are now going to construct codes from algebraic curves.

4.4 Codes from Algebraic Curves

Let χ be an algebraic curve over a finite field $\mathbb{F}_q$. Choose rational points $P_1, \ldots, P_n \in \chi$ and a divisor D of degree N with a support disjoint from $\{P_1, \ldots, P_n\}$. The functions in $L(D)$ are defined at the points P_j and we define a linear map

$$\Phi : L(D) \to \mathbb{F}_q^n, \quad f \mapsto (f(P_1), \ldots, f(P_n)).$$

The image of this map is a linear code of length n. If Φ is injective then the dimension of the code is $k = \dim L(D) \geq N - g + 1$.

We get the following result

Theorem 4.19. *Let $0 \leq N < n$. Then $\Phi(L(D))$ is an $[n, k, d]$-code with $k \geq N - g + 1$ and $d \geq n - N$.*

Proof. Recall that in general the minimum distance d of a linear code has the property that for every codeword c we have $w(c) < d \Rightarrow c = 0$, because $d = \min_{u \neq v} d(u, v) = \min_{u \neq 0} w(u)$.

Let us take a look at what happens if we want a function $f \in L(D)$ to vanish in at least $n - \delta$ points P_j, say $P_1, \ldots, P_{n-\delta}$. Then $f \in L(D - P_1 - P_2 - \ldots - P_{n-\delta})$. Now if we have $n - \delta < N$, then the degree of the above divisor is negative and we get $f = 0$. So if we know that f is non-zero in δ points with $\delta < n - N$, then $f = 0$. Therefore the minimum distance d satisfies $d \geq n - N$.

For $\delta = 0$ we see that the kernel of Φ is trivial; hence Φ is injective as long as $N < n$. This proves the theorem. □

Remark. Note that $\Phi(L(D))$ is well-defined, as for $f \in L(D)$, we have $v_{P_i}(f) \geq 0$ (i=1, …, n) because $\{P_1, \ldots P_n\} \cap \operatorname{supp} D = \emptyset$.

One of the most important classes of algebraic-geometric codes are the (geometric) Goppa codes. They are a special case of the above definition:

Definition 4.20. Let $P_1, \ldots, P_n$ be different rational points, define divisor $D = P_1 + \cdots + P_n$ and let G be a divisor with $\operatorname{supp} G \cap \operatorname{supp} D = \emptyset$. Then the *geometric Goppa code* $\mathcal{C}_{\mathcal{L}}(D, G)$ associated with the divisors D and G is defined by

$$\mathcal{C}_{\mathcal{L}}(D, G) := \{(f(P_1), \ldots, f(P_n)) \colon f \in L(G)\} \subseteq \mathbb{F}_q^n.$$

Asymptotically good Sequences of Codes

Definition 4.21. Let χ be an $[n,k,d]$-code. The quotient $R = k/n$ is called the *information rate* and $\delta = d/n$ denotes the *relative minimum distance.*

From (4.19) we know that

$$k + d \geq n + 1 - g,$$

hence

$$R + \delta \geq 1 - \frac{(g-1)}{n}.$$

Definition 4.22. A sequence of codes $(\chi_m)_{m \in \mathbb{N}}$ with parameters $[n_m, k_m, d_m]$ over a fixed finite field $\mathbb{F}_q$ is called *asymptotically good* if n_m tends to infinity while d_m/n_m and k_m/n_m tend to non-zero constants δ and R, respectively, for $m \rightarrow \infty$.

Now n cannot be larger than the number of rational points, so if we want to find asymptotically good codes we need curves having many rational points.

5 Decoding Algebraic-Geometric Codes

First we want to give a short overview of the historical developement of decoding algorithms for algebraic geometric codes. A far more detailed section can be found in Høholdt and Pellikaan [31].

Historical Outline

A first attempt to decode algebraic-geometric codes was made by Driencourt [14] for codes on elliptic curves. This algorithm corrects $\lfloor(\delta_\Gamma - 1)/4\rfloor$ errors, where δ_Γ is the *Goppa designed minimum distance*. At the end of the 80s Justesen, Elbrønd Jensen, Havemose and Høhold [28, 36] found a generalization of the decoding algorithm of Arimoto [2] and Peterson [55] for Reed-Solomon codes for algebraic-geometric codes on plane curves . This algorithm finds an *error-locator polynomial* in two variables which has the error positions among its zeros. This was generalized to arbitrary curves by Skorobogatov and Vlăduţ [72].

In his thesis, Porter [56] gave another decoding algorithm generalizing the solution of the *key equation* with *Euclid's algorithm* by Sugiyama, Kasahara, Hirasawa and Namekawa [76]. The correctness of the algorithm and the fact that it decodes $(\delta_\Gamma - 1)/2 - \sigma$ errors was shown in papers of Porter, Shen and Pellikaan [56, 57] and by Erhard [19, 18].

An effective algorithm which corrects $(\delta_\Gamma - 1)/2$ errors was given by Ehrhard [20]. The problem of finding the divisors $F_1, \dots, F_s$ in advance was circumvented by letting the algorithm find those divisors depending on the received word.

An elegant solution of the decoding problem for algebraic-geometric codes by a *majority vote* for *unknown syndromes* was proposed by Feng and Rao [22]. They showed in collaboration with Duursma [16, 15] that the algorithm corrects $(\delta_\Gamma - 1)/2$ errors. The origin of these methods stems from the decoding of cyclic codes beyond the BCH error-correcting capacity by Feng and Tzeng [24]. As a result of the majority voting it was noticed that sometimes one can even correct beyond half the Goppa designed minimum distance [23]. This was formalized by Kirfel and Pellikaan [38] who introduced the *Feng-Rao designed minimum distance* δ_{FR} for one-point codes.

The Berlekamp-Massey algorithm [4, 48] on *linear recurring relations* in one variable was generalized to several variables by Sakata [59, 60]. This algorithm was applied by Justesen, Larsen, Elbrønd Jensen and Høholdt [37] and Shen [70, 69] to get faster implementations of Krachkovskiis modified algorithm. Based on the Feng-Rao majority scheme from [22], majority voting was also incorporated by Sakata, Justesen, Madelung, Elbrønd Jensen and Høholdt [47, 62, 61, 63]. For a class of space curves, the method of [37] was generalized in [12], but here the algorithm does not correct all errors up to half the minimum distance.

BCH-, Reed-Solomon and some other codes can be decoded efficiently by using the Berlekamp-Massey algorithm, and it is still important in algebraic coding theory.

Generalized Berlekamp-Massey Decoding

We are going to show how to decode a general class of algebraic-geometric codes, the so-called *one-point codes*, up to half the Feng-Rao bound. In the first section we define the codes and give an overview of how the algorithm works. The next section deals with the algorithm in more detail, and finally the complexity will be calculated.

The algorithm is the modification from [61] of Sakata's algorithm [60].

5.1 The Codes and the Idea of the Algorithm

Let $\{P_1, \ldots, P_n, P_\infty\}$ be a set of $\mathbb{F}_q$-rational points on a nonsingular curve[4] of genus g defined over $\mathbb{F}_q$. One-point codes are defined as $\mathcal{C}_\mathcal{L}(D,G)^\perp$, where

$$D = P_1 + \cdots + P_n \quad \text{and} \quad G = mP_\infty.$$

The code $\mathcal{C}$ has length n, and for any $y \in \mathbb{F}_q^n$ we have

$$y \in \mathcal{C} \Leftrightarrow \sum_{j=1}^{n} f(P_j)y_j = 0 \quad \text{for all } f \in L(mP_\infty). \tag{3}$$

If $2g-2 < m < n$, the dimension of $\mathcal{C}$ is $k = n - m + g - 1$, and the minimum distance is lower-bounded by $d^* = m - 2g + 2$. If $m < 4g-2$ this estimate is improved by the Feng-Rao bound d_{FR}, which will be defined later. One has $d_{FR} \geq d^*$, with equality when $m \geq 4g-2$.

A number o_i is called a *nongap* for P_∞ if $L(o_iP_\infty) \neq L((o_i-1)P_\infty)$. In this case, there exists a function $\varphi_i \in L(o_iP_\infty)\backslash L((o_i-1)P_\infty)$. This means that φ_i has a pole of order o_i at P_∞ and no other poles.

As $\dim L(mP_\infty) = m-g+1$ and all the φ_i (from above) are linearly independent, $\{\varphi_i : i = 1, 2, \ldots, m-g+1\}$ forms a basis for the space $L(mP_\infty)$.

The nongap sequence, i.e. the sequence of possible pole orders at P_∞, forms a semigroup under addition. Let $a_1, \ldots, a_N$ be a minimal set of generators for this semigroup, and with $j = 1, \ldots, N$ let ψ_j be a function with pole order a_j at P_∞ and no other poles. To any vector $\alpha = (\alpha_1, \ldots, \alpha_n)$ of non-negative integers we define a function

$$f_\alpha = \prod_{j=1}^{N} \psi_j^{\alpha_j}. \tag{4}$$

This function has a pole only at P_∞. The order of this pole is denoted by $O(\alpha)$, and we have

$$O(\alpha) = \sum_{j=1}^{N} \alpha_j a_j.$$

The set of functions f_α where $O(\alpha) \leq m$ span the space $L(mP_\infty)$. These functions are, however, not necessarily independent, since if $O(\alpha) = O(\alpha')$ then

$$f_\alpha = cf_{\alpha'} + g, \quad \text{where } c \in \mathbb{F}_q \quad \text{and} \quad O_{P_\infty}(g) < O(\alpha). \tag{5}$$

[4] A nonsingular curve does not contain points with all partial derivatives equal to zero

An important concept in decoding is the *syndrome* of a vector. Let $y \in \mathbb{F}_q^n$; with each function f_α we associate the syndrome $S_\alpha(y)$ defined by

$$S_\alpha(y) = \sum_{j=1}^{n} f_\alpha(P_j) y_j. \tag{6}$$

As the set $\{f_\alpha \colon O(\alpha) \leq m\}$ spans $L(mP_\infty)$, it follows with (3) that

$$y \in \mathcal{C} \Leftrightarrow S_\alpha(y) = 0 \quad \text{for all } \alpha \text{ with } O(\alpha) \leq m.$$

In the decoding situation we receive a vector r which is the sum of an unknown codeword c and an unknown error vector e. We therefore have $S_\alpha(e) = S_\alpha(r)$ for all α with $O(\alpha) \leq m$, so the syndromes $S_\alpha(e)$ can be calculated directly from the received word if $O(\alpha) \leq m$. The problem is to recover the vector e from the known terms $S_\alpha(e)$.

In [37] this is done reasonably efficient by considering recursions among the syndromes and from such equations determining an error locator, that is, a function which points out the positions where the coordinates in e are different from zero. Unfortunately, this procedure does not correct errors up to half the minimum distance.

Another approach is first to determine *all* syndromes $S_\alpha(e)$, $0 \leq \alpha_j \leq q-1$, $i = 1, \ldots, N$. In the next section, we will explain how this is done. If we know all the syndromes then for each point P_l we can form the sum

$$\sum_\alpha S_\alpha(e) \prod_{s=1}^{N} \psi_s^{-\alpha_s}(P_l), \tag{7}$$

where the summation is over all vectors α with $1 \leq \alpha_s \leq q-1$ and $s = 1, \ldots, N$. In the following we will see that by inserting (6) and (4) we get

$$\begin{aligned}
\sum_\alpha S_\alpha(e) \prod_{s=1}^{N} \psi_s^{-\alpha_s}(P_l) &= \sum_\alpha \sum_{j=1}^{n} \prod_{s=1}^{N} \psi_s^{\alpha_s}(P_j) \psi_s^{-\alpha_s}(P_l) e_j \\
&= \sum_{j=1}^{n} e_j \prod_{s=1}^{N} \sum_\alpha \left(\frac{\psi_s(P_j)}{\psi_s(P_l)} \right)^{\alpha_s} \\
&= (-1)^N e_l
\end{aligned} \tag{8}$$

and hence e_l can be calculated.

The above equalities need some remarks: First, we can change the order of summation and multiplication because we sum over all possible α with $1 \leq a_s \leq q-1$ and $s = 1, \ldots, N$.

Secondly, if $\psi_s(P_j) \neq \psi_s(P_l)$, then

$$\begin{aligned}
\sum_{\alpha_s=1}^{q-1} \left(\frac{\psi_s(P_j)}{\psi_s(P_l)} \right)^{\alpha_s} &= \frac{\psi_s(P_j)}{\psi_s(P_l)} \sum_{\alpha_s=0}^{q-2} \left(\frac{\psi_s(P_j)}{\psi_s(P_l)} \right)^{\alpha_s} \\
&= \frac{\psi_s(P_j)}{\psi_s(P_l)} \cdot \frac{1 - \left(\frac{\psi_s(P_j)}{\psi_s(P_l)} \right)^{q-1}}{1 - \frac{\psi_s(P_j)}{\psi_s(P_l)}} \\
&= 0,
\end{aligned}$$

as $a^{q-1} \equiv 1 \mod q$ for all a by Fermat's Little Theorem. If $j \neq l$, then for at least one s we have $\psi_s(P_j) \neq \psi_s(P_l)$; because otherwise $f_\alpha(P_j) = f_\alpha(P_l)$ for each α. Consequently, there is a codeword of weight 2, and we do not consider such codes.

The third remark is that in the calculation we have used $\psi_s(P_l) \neq 0$ for all $s = 1, \ldots, N$. If this is not the case, the calculations should be slightly modified as follows, but in any case, knowing all syndromes, we can calculate the error vector.

Now we treat the case where $\psi_s(P_l) = 0$ for some, but not all, $s \in \{1, \ldots, N\}$. Among all points with this property we define a partial order given by $P < Q$ if and only if $\psi_i(P) = \psi_i(Q)$ for all i where $\psi_i(Q) \neq 0$.

Let P_l be a point where $\psi_{i_1}(P_l) \neq 0, \ldots, \psi_{i_r}(P_l) \neq 0$ and $\psi_j(P_l) = 0$ for $j \in T = \{1, \ldots, N\} \backslash \{i_1, \ldots, i_r\}$. We form the sum

$$\sum_\alpha S_\alpha \prod_{s=1}^{r} \psi_{i_s}^{-\alpha_{i_s}}(P_l)$$

where the summation is over all vectors α with $\alpha_j = 0$ if $j \in T$ and $1 \leq \alpha_j \leq q-1$ otherwise. This sum equals

$$\sum_{j=1}^{n} e_j \prod_{s=1}^{r} \sum_{\alpha_{i_s}=1}^{q-1} \left(\frac{\psi_{i_s}(P_j)}{\psi_{i_s}(P_l)} \right)^{\alpha_{i_s}},$$

which we write as

$$\sum_{j=1}^{n} e_j c_j.$$

If $\psi_{i_s}(P_j) = \psi_{i_s}(P_l)$ for $s = 1, \ldots, r$, we have $c_j = (-1)^r$, and otherwise we have $c_j = 0$. Consequently, $c_j \neq 0$ if and only if $P_j < P_l$, and therefore the sum is

$$(-1)^r \left(e_l + \sum_{P_j < P_l} e_j \right).$$

Now, if the point P_l is minimal with respect to the partial order, we retrieve e_l directly in this way. So if we do the calculations according to the partial order starting with minimal elements, the terms in the above expression are all known except e_l, which can therefore be calculated.

The only situation left is where $\psi_s(P_l) = 0$ for all $s = 1, \ldots, N$. There can be at most one such point Q (because otherwise the minimum distance is 2). We start by calculating all other error values e_P, where $P \neq Q$. Since

$$S_0 = \sum_{j=1}^{n} e_j,$$

it is easy to calculate e_Q.

5.2 The Algorithm

The basic idea is that, from a set of known syndromes $S_\alpha(e)$ with $O(\alpha) \leq m$, we want to find $S_\alpha(e)$, for $O(\alpha) = m' > m$.

The algorithm gets as input an N-dimensional array of elements from $\mathbb{F}_q$ and outputs a so-called minimal set of polynomials corresponding to linear recurring relations satisfied by the array. In order to describe the algorithm, some notation from [60] is required.

Let Σ_0 be defined as the set of all N-tuples of non-negative integers, that is $\Sigma_0 = \mathbb{N}_0^N$. For any subset $\Gamma \subseteq \Sigma_0$, an *array* over $\mathbb{F}_q$ is a mapping $u : \Gamma \to \mathbb{F}_q$, which is written $u = (u_\alpha)$, where $u_\alpha = u(\alpha)$, $\alpha \in \Gamma$, is the *value* of the array of the point α.

We need a well-ordering of the elements in Σ_0.

Definition 5.1. A well-ordering $<_T$ of the elements in Σ_0 is called *admissible* if the following holds:
1) For any $\alpha \in \Sigma_0 : (0, \ldots, 0) \leq_T \alpha$.
2) For any $\alpha, \beta, \gamma \in \Sigma_0$: if $\alpha <_T \beta$, then $\alpha + \gamma <_T \beta + \gamma$.

Remark. A total order satisfying (1) and (2) is sometimes called a *monomial of reduction order* in the Gröbner-basis literature.

Example 5.2. *The* lexicographic order $<_L$ *is defined by*

$$(p_1, \ldots, p_N) <_L (q_1, \ldots, q_N)$$

if and only if $p_1 = q_1, \ldots, p_{l-1} = q_{l-1}$ *and* $p_l < q_l$ *for some* $1 \leq l \leq N$.
The total degree lexicographic order $<_T$ *is defined by*

$$p = (p_1, \ldots, p_N) <_T q = (q_1, \ldots, q_N)$$

if and only if $O(p) < O(q)$ *or* $(O(p) = O(q)$ *and* $p <_L q)$.

Corresponding to the code described above, we choose the total degree lexicographic order. This also gives an ordering of the functions f_α and the syndromes $S_\alpha(e)$. It should be mentioned that Sakata's algorithm works with any admissible ordering of Σ_0.

It is convenient to represent linear recurring relations by N-variate polynomials $\sigma \in \mathbb{F}_q[x] = \mathbb{F}_q[\psi_1, \ldots, \psi_N]$. Any such polynomial can be written as

$$\sigma = \sum_{q \in \Gamma_\sigma} \sigma_q x^q,$$

where $x^q = \psi_1^{q_1} \cdots \psi_N^{q_N}$ and Γ_σ is a finite subset of Σ_0 such that $\sigma_q \neq 0$ for $q \in \Gamma_\sigma$. The maximal element in Γ_σ with respect to the total order $<_T$ is called the *degree* of σ and is written $\mathrm{Deg}(\sigma)$.

A polynomial σ is said to be *valid at a point* p for an array u if $p \geq s = \mathrm{Deg}(\sigma)$ and

$$\sum_{q \in \Gamma_\sigma} \sigma_q u_{q+p-s} = 0. \tag{9}$$

Here $\geq$ is the natural partial order on Σ_0 defined by $p \geq q$ if and only if $p_i \geq q_i$ for all $i = 1, \dots, N$. Moreover, here and in the following we assume that Γ is of the form

$$\Gamma = \{x \in \Sigma_0 \colon x <_T l\},$$

and write $u = u^l$ for the corresponding array.

A polynomial σ is said to be *valid* for an array u if σ is valid at all points $p \in u^l$ where $s \leq p$. The set of valid polynomials for an array u is denoted VALPOL(u).

To understand the whole setup better, we will explain where this leads us to. As mentioned above consider the array of known syndromes $S_\alpha(e)$ where $O(\alpha) \leq m$. Inserting (6) in (9) we get

$$\begin{aligned}\sum_{q \in \Gamma_\sigma} \sigma_q S_{q+p-s}(e) &= \sum_{q \in \Gamma_\sigma} \sigma_q \sum_{j \in E} f_{q+p-s}(P_j) e_j \\ &= \sum_{j \in E} e_j f_{p-s}(P_j) \sum_{q \in \Gamma_\sigma} \sigma_q f_q(P_j), \end{aligned} \tag{10}$$

where $E = \{j_1, \dots, j_t\}$ denotes the positions for which the error vector is non-zero. It follows from this that if the function

$$f = \sum_{q \in \Gamma_\sigma} \sigma_q f_q$$

is zero at all error points $P_{j_1}, \dots, P_{j_t}$, then the polynomial σ satisfies all possible recurring relations (9) for that polynomial and the array considered.

Let us return to the general situation where we consider an array $u = u^l$.

Definition 5.3. For an array u over $\mathbb{F}_q$, a *minimal polynomial set* is a finite subset $\mathcal{F}$ of $\mathbb{F}_q[x]$ such that

1) $\mathcal{F} \subseteq$ VALPOL(u)
2) Let $\mathcal{S} = \{\mathrm{Deg}(\sigma) \colon \sigma \in \mathcal{F}\}$, then for any $\sigma < \tau$, $\sigma \in \mathcal{S}$ implies $\tau \notin \mathcal{S}$
3) If $g \in$ VALPOL(u), then there exists a $\sigma \in \mathcal{S}$ such that $\sigma \leq \mathrm{Deg}(g)$.

Remark. Let $\Delta = \Delta(\mathcal{F})$ be the complement of $\{\tau \in \Sigma_0 \colon \sigma \leq \tau \text{ for some } \sigma \in \mathcal{S}\}$ in Σ_0. The third condition can now be rephrased by saying that there exists no polynomial $g \in$ VALPOL(u) such that $\mathrm{Deg}(g) \in \Delta$. This set Δ is called the *delta set* or the *footprint* [7] of $\mathcal{F}$. It follows that the word “minimal” in the term “minimal polynomial set” refers to the degrees of the polynomials in the set.

The *algorithm by BMS (Berlekamp-Massey-Sakata)* takes as input the elements of an array $u = u^l$ and outputs a minimal polynomial set for the array. The algorithm considers the elements of the array step by step. At each step one has a minimal polynomial set $\mathcal{F}$ for the part of the array seen so far. When the next element of the array is taken into consideration, the algorithm starts to check if the polynomials $f \in \mathcal{F}$ are still valid for the new array. If this is not the case, they are updated and a new polynomial set and a new Δ-set are produced.

The following Lemma from [60, Lemma 2] is essential for the whole process:

Lemma 5.4. *Let* $\mathrm{Deg}(\sigma) = s$*. If* $\sigma \in \mathrm{VALPOL}(u^q)$ *and* $\sigma \notin \mathrm{VALPOL}(q^{q+1})$*, then there exists no polynomial* $g \in \mathrm{VALPOL}(u^{q+1})$ *with* $\mathrm{Deg}(g) \leq q - s$*.*

Here $q + 1$ *denotes the next greater point of* q *with respect to the total order.*

Let us return to the situation where we know all syndromes $S_\alpha(e)$ with $O(\alpha) \leq m$ and we want to find $S_\alpha(e)$ with $O(\alpha) > m$. There may be many syndromes corresponding to the same pole order. But if $O(\alpha) = O(\alpha')$, we have an identity (5) between f_α and $f_{\alpha'}$, and hence also an identity for the syndromes

$$S_\alpha = cS_{\alpha'} + \sum_{O(\beta)<O(\alpha)} c_\beta S_\beta. \tag{11}$$

We want to distinguish between functions or syndromes, which are dependent — in the sense of (5) or (11) — and those, which are independent.

To this end, we choose a set $\Sigma' \subseteq \Sigma_0$ such that Σ' contains exactly one element x corresponding to each poleorder $O(x)$.

In the algorithm we now only consider those polynomials whose degree belongs to Σ'. This is possible according to (5). As a consequence we use Σ' instead of Σ_0 in the definition of $\Delta = \Delta(\mathcal{F})$, which means that mutually distinct points in Δ correspond to functions with mutually distinct pole orders. Such functions are independent, a fact used in the next result, which like Lemma 5.4, is essential for the whole setup.

Lemma 5.5. *Suppose that the number of errors that occurred is equal to* t*. Then, in each step of the algorithm the number of points in the* Δ*-set is at most* t*.*

Proof. Let R denote the ring of functions which have no poles outside P_∞, and let $I \subseteq R$ be the ideal of those functions, which are zero at the error points $P_{j_1}, \ldots, P_{j_t}$. Then the dimension of R/I, as a vector space over $\mathbb{F}_q$, is equal to t. Now, for each $a \in \Delta$ we take a polynomial σ_a with $\mathrm{Deg}(\sigma_a) = a$, the corresponding function $g_a \in R$ and the image $[g_a] \in R/I$. Here $g_a \notin I$, because otherwise the expressions (10) were zero, and hence σ_a was valid. The same holds for any linear combination of functions g_a. Therefore, the number of elements in Δ is at most the dimension of R/I, that is, at most t. □

Let us return to the decoding situation. Let $\gamma \in \Sigma'$ satisfy $O(\gamma) = m'$. Put $\gamma^{(0)} = \gamma$ and let $\gamma^{(1)}, \gamma^{(2)}, \ldots$ be all other elements of Σ_0 with pole order m'. By $\mathcal{F} = \{\sigma^{(1)}, \ldots, \sigma^{(k)}\}$ we denote a minimal polynomial set for the array s with values s_α in the domain $\Gamma = \{\alpha\colon O(\alpha) < m'\}$, where $\mathrm{Deg}(\sigma^{(i)}) \in \Sigma'$. Suppose without loss of generality that all $\sigma^{(i)}$ have leading coefficient 1.

Let $\mathrm{Deg}(\sigma^{(i)}) = s^{(i)}$ and suppose that $s^{(i)} \leq \gamma(j)$, then it is possible to calculate

$$S'_{\gamma^{(j)}} = -\sum_{q\in\Gamma^i} \sigma_q S_{q+\gamma^{(j)}-s^{(i)}}, \tag{12}$$

where $\Gamma^i = \Gamma_{\sigma^{(i)}} \backslash s^{(i)}$. If $\sigma^{(i)}$ is valid at $\gamma^{(j)}$, then (9) holds, that is,

$$S_{\gamma^{(j)}} - S'_{\gamma^{(j)}} = 0.$$

From this equation we can calculate $S_{\gamma^{(j)}}$ and then S_γ is determined by (11).

If $\sigma^{(i)}$ is not valid at the point $\gamma^{(j)}$, that is, if (12) does not hold for the correct value of $S_{\gamma^{(j)}}$, then $\sigma^{(i)}$ must be updated. This updating will increase the size of the Δ-set, and we can use Lemma 5.4 to estimate how much the Δ-set is increased by.

To state the results precisely, set

$$K(\gamma) = \{x \in \Sigma' \colon \exists \gamma^{(j)} : x \leq \gamma^{(j)} \wedge \gamma^{(j)} - x \in \Sigma'\}.$$

For each $\sigma^{(i)}$ with $\mathrm{Deg}(\sigma^{(i)}) = s^{(i)}$, we check if there is a $\gamma^{(j)}$ with $\gamma^{(j)} \geq s^{(i)}$ and $\gamma^{(j)} - s^{(i)} \in \Sigma'$. If this is the case, use (12) and (11) to predict the value of s_γ and set

$$K_i = \{x \in K(\gamma) \colon x \leq \gamma^{(j)} - s^{(i)}\}.$$

If such a $\gamma^{(j)}$ does not exist, then $\sigma^{(i)}$ is not used to find the correct value of S_γ.

Let v_i denote the value of S_γ predicted by $\sigma^{(i)}$, if this situation occurs. If v_i turns out to be wrong, then according to Lemma 5.4, all the points in K_i belong to the new Δ-set. Therefore, if we set

$$K_i' = K_i \backslash \Delta, \tag{13}$$

the Δ-set increases at least by K_i', if v_i is not the correct value.

Let $w_1, \ldots, w_p$ be the *different* predictions v_i for S_γ obtained by the above method, and for each $j = 1, \ldots, p$ let L_j denote the union of the sets from (13) for which $v_i = w_j$:

$$L_j = \bigcup_{\substack{i=1 \\ v_i = w_j}}^{k} K_i \backslash \Delta$$

The *Feng-Rao distance*, d_{FR}, for this code is defined by

$$d_{FR} = \min_{\substack{\gamma \in \Sigma' \\ O(\gamma) > m}} |K(\gamma)|.$$

The main theorem provides a very simple way to find the correct value of the next syndrome S_γ; it goes back to an idea by Feng and Rao.

Theorem 5.6. *Suppose that the number t of errors satisfies*

$$t \leq \left\lfloor \frac{d_{FR} - 1}{2} \right\rfloor$$

and let $l \in \{1, \ldots, p\}$ be the number for which $|L_l|$ is maximal. Then for the syndrome S_γ we have

$$S_\gamma = w_l.$$

For a proof see [61].

5.3 The Complexity

This section gives a short desciption of how the complexity of the BMS algorithm is derived. The complete decoding algorithm can be described as follows:

1) Calculate the syndromes S_α, where $O(\alpha) \leq m$, using (5) and (6).

2) Use Sakata's algorithm to find a reduced minimal polynomial set for the array of known syndromes. By *reduced* we mean that the degrees of all polynomials belong to Σ'.

3) Use Theorem 5.6 to find S_γ, where $O(\gamma) = m + 1$ and $\gamma \in \Sigma'$.

4) Calculate all $S_{\gamma(i)}$ using (11).

Repeat step 2) to step 4) until all syndromes S_γ, where $O(\gamma) \leq d_{FR} + 4g$, are known (which means that $2g$ new syndromes must be calculated).

5) Calculate the remaining syndromes using (11) and (12) with polynomials from the last minimal set.

6) Calculate the error values using (8).

It is convenient to distinguish between independent and dependent syndromes. For $\alpha \in \Sigma'$ we call S_α an independent syndrome. All the dependent syndromes can be calculated from the independent syndromes by simple linear combinations, as can be seen in (11).

The number of terms on the right-hand side in (11) is at most $r = O(\alpha)$. So if $A(r)$ denotes the number of syndromes of order r, then the complexity of finding the dependent syndromes of order r is $rA(r)$. Hence we first focus on the independent syndromes, and then find the complexity related to the dependent syndromes.

1) There are $m - g + 1$ independent syndromes S_α with $O(\alpha) \leq m$, and the calculation costs $(m - g + 1) \cdot 2n$ operations.

2) The number of polynomials in a reduced minimal set is bounded above by the smallest pole order, denoted a_1. From Sakata's results [60, p. 228] it follows that one iteration of the algorithm has complexity $\mathcal{O}(a_1(r - g + 1))$, where r is the pole order in question. The complexity of finding a reduced minimal polynomial set for the array of known syndromes is $\mathcal{O}(a_1(m - g + 1)^2)$.

3) To calculate the candidate values for S_γ, where $O(\gamma) = m + 1$, costs at most $a_1(m - g)$ operations. Moreover, we must find the number of elements in the sets K_i', which costs at most $a_1 \cdot d$ operations, where $d = d_{FR}$.

We must repeat the calculation of new syndromes and the update of the reduced minimal set up to pole order $d + 4g$. The complexity for this is

$$\mathcal{O}((d - 4g - m)a_1(m - g)) + \mathcal{O}((d - 4g - m)a_1 d) + \mathcal{O}(a_1(d + 3g + 1)^2)$$

Using the upper bound n for both m and d, the complexity of the steps considered so far is at most $\mathcal{O}(a_1 \cdot n^2)$.

4) and 5) Calculating all dependent syndromes of order r costs $rA(r)$ operations, as stated above. By summing up $rA(r)$ over all pole orders, we get an upper bound on the complexity. If $r = x_1a_1 + \cdots + x_Na_N$ then

$$\begin{aligned}\sum_r A(r)\cdot r &= \sum_x (x_1a_1 + \cdots + x_Na_N)\\ &= \sum_{x_1=0}^{q-1} \cdots \sum_{x_N=0}^{q-1} (x_1a_1 + \cdots + x_Na_N)\\ &= q^{N-1}\sum_{i=1}^{N} a_i \sum_{x_i=0}^{q-1} x_i\end{aligned}$$

and the magnitude of this is $q^{N+1}(a_1 + \cdots + a_N)$.

6) The magnitude of calculating the error values using (7) is $n \cdot q^N \cdot N$ operations.

Altogether, the complexity of this algorithm is upper-bounded by

$$\mathcal{O}(a_1 \cdot n^2) + \mathcal{O}(q^{N+1}(a_1 + \cdots + a_N)) + \mathcal{O}(n \cdot N \cdot q^N) \tag{14}$$

Example 5.7. *Let us consider the curve in the affine 3-space over $\mathbb{F}_q$, where $q = p^2$, defined by*

$$y^{p+1} = x^p + x \quad z^{p+1} = -xy^p - yx^p - 1.$$

It follows from [79] that if $r \equiv 1 \mod 3$, the curve has $(p^2-1)^2$ $\mathbb{F}_q$-rational points and has genus $p^3 + p^2 - p$. At P_∞, the common pole of x, y and z, the functions x, y and z have pole orders $(p+1)^2$, $p(p+1)$, and $p(p+2)$, respectively. If we express all the terms in (14) *using the code length n, we get*

$$\mathcal{O}(n^{1/2} \cdot n^2) + \mathcal{O}(n^2 \cdot 3 \cdot n^{1/2}) + \mathcal{O}(n \cdot 3 \cdot n^{3/2}),$$

so in this case the complexity is $\mathcal{O}(n^{5/2})$.

5.4 Pseudo-Code of the Algorithm

Let $P_1, P_2, \ldots, P_n, P_\infty$ be $\mathbb{F}_q$-rational points on a nonsingular curve χ of genus g defined over $\mathbb{F}_q$. We consider an algebraic geometry code $\mathcal{C}_m$ of type $\mathcal{C}_l(D,G)^\perp$, where $D = P_1+P_2+\cdots+P_n$ and $G = mP_\infty$.

Now we need some more notation:

Let R denote the ring of all rational functions on the curve χ with poles only at P_∞, that is,

$$R = \bigcup_{a=0}^{\infty} L(aP_\infty).$$

For $f \in R$ we let $\rho(f)$ denote the poleorder of f at P_∞.

Similar to the above, we define syndromes also for functions. Let $f \in R$ and $y \in \mathbb{F}_q^n$, then the syndrome $S_y(f)$ is defined as

$$S_y(f) = \sum_{i=1}^{n} y_i f(P_i).$$

For $S_e(f)$, where e is the error vector, we sometimes omit the subscript e.

Definition 5.8. Let φ_i, $i = 1, 2, \ldots, m - g + 1$ be a basis of the space $L(mP_\infty)$ and o_i the nongaps for P_∞. We define the *span* of an element $f \in R$ by $\operatorname{span}(f) = o_i$ if $S(f\varphi_i) \neq 0$ but $S(f\varphi_l) = 0$ for all $l < i$.

The decoding algorithm uses algorithm 1 which consists of two main parts. The first part, steps 1-3, is an iterative procedure that, based on the syndromes $S(\varphi_r)$, where $o_r < m$, calculates two sets of functions

$$\begin{aligned} F_M &= \{f \in R \colon S(f\varphi_j) = 0 \text{ for all } j, \rho(f) + o_j \leq m\}, \\ G_M &= \{g \in R \colon S(g\varphi_i) \neq 0 \text{ for some } i, \rho(g) + o_i \leq m\}, \end{aligned}$$

where $M = m - g + 1$. Furthermore, the following set of poleorders is calculated

$$\Delta_M = \{\rho(g) \colon g \in G_M\} = \{\operatorname{span}(g) \colon g \in G_M\}.$$

The second part, step 4, uses for $M' \geq M$ the obtained sets $F_{M'}$, $G_{M'}$ and $\Delta_{M'}$ to determine $S(f)$ where $\rho(f) = m' + 1$, $m' \geq m$ by a voting procedure. This algorithm solves the decoding problem when $\tau \leq \lfloor (d_{FR} - 1)/2 \rfloor$, where τ is the Hamming weight of the error vector e (see [51], [61] or [34]).

Algorithm 1

```
**Input:** $S(\varphi_i)$, $i \leq m$.
**Initialization:** $F_0 = \{1\}$, $\Delta_0 = \emptyset$, $G_0 = \emptyset$.
At order $o_{l+1}$, $l = 0, \dots, m+g-1$ let $A = \emptyset$ and iterate the following

**1.** For each $f \in F_l$
**if** $\rho(f) \not\leq_T o_{l+1}$ or $S(f\varphi_j) = 0$ where $o_j = o_{l+1} - \rho(f)$ **then**
    $f \in F_{l+1}$
**else**
    **if** $o_{l+1} - \rho(f) \leq_T \operatorname{span}(g)$ for some $g \in G_l$ **then**
        $\hat{f} = f + \beta g \varphi_j \in F_{l+1}$ where $\operatorname{span}(g) - (o_{l+1} - \rho(f)) = o_j$ and $\beta \in \mathbb{F}_q^*$
    **else**
        $f \in G_{l+1}$ and $f \in A$
    **end if**
**end if**

**2.** For each $g \in G_l$
**if** $\operatorname{span}(g) \not\leq_T \operatorname{span}(f)$ for some $f \in A$ **then**
    $g \in G_{l+1}$
    $\Delta_{l+1} = \{o \colon o \leq_T \operatorname{span}(g), g \in G_{l+1}\}$
**end if**

**3.** For each $o \in \Sigma' \backslash \Delta_{l+1}$ which is minimal with respect to $\leq_T$ and for which $o = \rho(f) + o_i$,
$f \in A$, $o_i > 0$ holds
**if** $o_{l+1} - o \leq_T \operatorname{span}(g)$ for some $g \in G_l$ **then**
    $\hat{f} = \varphi_i f + \beta \varphi_j g \in F_{l+1}$ where $\operatorname{span}(g) - (o_{l+1} - o) = o_j$, $\beta \in \mathbb{F}_q^*$
**else**
    $f\varphi_i \in F_{l+1}$
**end if**
```

To explain the voting procedure we need the following notation:

$$\begin{aligned} &\Sigma_i = \rho(R) \backslash \Delta_i, \\ &\Gamma_i = \{o \in \Sigma_{i-1} \colon o \leq_T o_i \text{ and } o_i - o \in \Sigma_{i-1}\} \end{aligned}$$

Suppose we have $F_{M'}, G_{M'}$, and $\Delta_{M'}$, and, therefore, also $\Sigma_{M'}$ and $\Gamma_{M'+1}$ for some $M' \geq M$.

```
**4.** For each $o \in \Gamma_{M'+1}$
Choose $o_s \in \Sigma_{M'}$ minimal with respect to $\leq_T$ such that $o_s \leq_T o$.
Select $\omega \in \mathbb{F}_q^*$ such that $g_o = \varphi_{M'+1} + \omega F(o_s) \varphi_{M'+1-s}$ satisfies $\rho(g_o) < o_{M'+1}$ where
$o_{M'+1} = m' + 1$ and $F(o_s)$ denotes the element of $F_{M'}$ of order $o_s$.
Let the vote by $g_o$ for $S(\varphi_{M'+1})$ be $S(g_o)$ and set

    $S(\varphi_{M'+1}) = \text{majority}\{S(g_o)\}$.
```

The complete decoding algorithm can now be given:

1) Calculate the syndromes $S_e(f) = S_y(f)$, $\rho(f) \leq m$ from the received word $y = c + e$.

2) Use algorithm 1 to determine the remaining syndromes $S_e(f)$ where $m < \rho(f) < 2(\tau + 2g) - 1$.

3) Calculate the error values using (8).

5.5 Remarks about the BMS Algorithm

In [64] Elbrønd Jensen, Høholdt, Leonard and Sakata showed how to decode algebraic geometry codes if the received word contains errors and erasures[5]. Their approach corresponds to the so-called nonstandard approach to extend the Berlekamp-Massey algorithm to erasures and errors. It produces a basis for the erasure-locator ideal, which can then be used as a seed to produce a basis for the errata (that is, erasure- and error-) locators. This extension requires only a small modification of the BMS algorithm.

Let t denote the number of errors and let τ denote the number of erasures, then their algorithm produces all the needed syndromes provided that $2t + \tau < d_{FR}$.

Michael E. O'Sullivan generalized the BMS algorithm to a broad class of rings [52]. The key concept for the generalization is the existence of an order function, a map from the ring to the nonnegative integers which determines a filtration of the ring with one-dimensional quotients. He also derives an improved bound for decoding based on the geometry of the error locations rather than the total number of errors.

Maria Bras-Amorós and Michael E. O'Sullivan investigated the error correction capability of the BMS algorithm [9]. If the errors are in general positions, the algorithm can often decode far more that $(d_{min} - 1)/2$ errors. They give a precise characterization of the error correction capability of the BMS algorithm and extend the concept behind Feng and Rao's improved codes to the decoding of errors in general position.

[5] An erasure is like an error with known error position.

6 A McEliece Cryptosystem using AG Codes

In this section we will discuss how to construct a McEliece cryptosystem using algebraic-geometric codes. Our aim is to minimize the size of the public key while keeping the security high enough for practical applications. We will compare the results with the algebraic-geometric code developed by Janwa and Moreno, and with different Goppa codes.

6.1 Review of Janwa and Moreno's Construction

In [33] Janwa and Moreno proposed to use algebraic-geometric codes for the McEliece cryptosystem. Citing results by Serre ([67] and [68]), that a certain curve of genus 2 exists over $\mathbb{F}_{2^7}$ with 172 rational points, they showed that this curve can be used to construct a $[171, 109, 61]$ code over $\mathbb{F}_{2^7}$.

Unfortunately, this construction has two major disadvantages. Firstly, it is not clear how to find this curve. In some cases the search for such a curve might be easy, but in general one wants to have a 'recipe' to construct codes with specific parameters, and not having to search for certain algebraic curves first. Secondly, this code is not secure enough for today's standards. The work factor to decypher a message with the original information-set-decoding-attack (Janwa and Moreno's calculations used this attack) is about the same as for the $[1024, 524, 101]$ Goppa code, which is not secure enough. Using the general information-set-decoding-attack the work factor is even less than for the Goppa code (see the table below).

For the first reason, we proposed to use the Generalized Srivastava codes. They can be defined much easier and have good bounds on their parameters. To overcome the second problem we tried to minimize the size of the public key while keeping the work factor greater than 2^{80}, which should provide high enough security.

Srivastava codes have even more good properties. According to D.V. Sarwate [65, Corollary 2] the class of alternant codes of block length n can be decoded using $\mathcal{O}(n \log^2 n)$ arithmetic operations, which is the same complexity as for Goppa codes. Thus, as Srivastava codes are in fact alternant codes, fast decoding algorithms exist.

As explained above, some subclasses of alternant codes would be an insecure choice for the McEliece cryptosystem: GRS codes [71], Goppa codes derived from binary generator polynomials [45] and, for the same reason, BCH codes [40].

This was another reason to choose Generalized Srivastava codes, because they have no intersection with those classes of insecure codes.

6.2 Generalized Srivastava Codes

We are going to describe the construction and some properties of the class of Generalized Srivastava codes. As a reference see [46], [30] or [77].

(Generalized) Srivastava codes are linear codes, which implies that they are also algebraic-geometric codes. This comes from the fact that Pellikaan, Shen and van Wee [54] proved that *every* linear code is, according to our definition, algebraic-geometric. For practical applications though, it is usually easier to describe them in terms of generator and/or check matrices than by curves and divisors. We therefore chose the former representation.

Let $\alpha_1, \dots, \alpha_n, w_1, \dots, w_s$ be $n+s$ distinct elements of $\mathbb{F}_q^m$ and $z_1, \dots, z_n$ be non-zero elements of $\mathbb{F}_q^m$. Then the generalized Srivastava code of length n has parity check matrix

$$H = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_s \end{bmatrix},$$

where

$$H_l = \begin{bmatrix} \frac{z_1}{\alpha_1 - w_l} & \frac{z_2}{\alpha_2 - w_l} & \cdots & \frac{z_n}{\alpha_n - w_l} \\ \frac{z_1}{(\alpha_1 - w_l)^2} & \frac{z_2}{(\alpha_2 - w_l)^2} & \cdots & \frac{z_n}{(\alpha_n - w_l)^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{z_1}{(\alpha_1 - w_l)^t} & \frac{z_2}{(\alpha_2 - w_l)^t} & \cdots & \frac{z_n}{(\alpha_n - w_l)^t} \end{bmatrix}$$

for some $t \geq 1$.

The original Srivastava codes are the special case $t = 1$, $z_i = \alpha_i^\mu$ for some μ and therefore have parity check matrix

$$H = \begin{bmatrix} \frac{\alpha_1^\mu}{\alpha_1 - w_1} & \cdots & \frac{\alpha_n^\mu}{\alpha_n - w_1} \\ \vdots & \ddots & \vdots \\ \frac{\alpha_1^\mu}{\alpha_1 - w_s} & \cdots & \frac{\alpha_n^\mu}{\alpha_n - w_s} \end{bmatrix}.$$

Since there are s w_i's, there can be at most $(q^m - s)$ α_i's, so the length of a generalized Srivastava code is at most $q^m - s$.

If $\alpha_1, \dots, \alpha_n$ are chosen to be all the elements of $\mathbb{F}_{q^m}$ except the w_i's, then $n = q^m - s$ and the codes are called *primitive* (by analogy with BCH codes).

Since it is an alternant code, a generalized Srivastava code has $k \geq n - mst$ and $d \geq st + 1$ (see [46]).

Hence a Srivastava code with above parameters is an $[n, k \geq n - mst, d \geq st + 1]$ code over $\mathbb{F}_q$.

6.3 Proposed Parameters

Although there are no further contraints on the parameters we want to introduce two for the following reasons:

For $t = 1$, every Srivastava code is a Goppa code (see [46, p. 359]). These have been studied for a long time now and have the known advantages and disadvantages. They are a secure choice for the McEliece cryptosystem, but they produce very large public keys, which makes them useless for most practical applications.

Choosing $s = 1$ gives the parity check matrix a much more regular structure. Similar to GRS codes, this might prove to be an insecure choice.

We therefore want t and s both to be greater than 1.

It is also reasonable to take $m = 1$, because the resulting codes are MDS codes, and also because the information rate $R = k/n$ decreases with increasing m.

We want the work factor for the generalized information-set-decoding attack by Lee/Brickell to be at least 2^{80} to provide high enough security.

Optimizing the parameters we found $s = 8$, $t = 11$ and $q = 2^8$ a good choice, resulting in a $[248, 160, 89]$ code over $\mathbb{F}_{2^8}$.

We compare this code to the $[2048, 1278, 141]$ and $[2048, 1608, 81]$ Goppa codes and the $[171, 109, 61]$ algebraic-geometric code developed by Janwa and Moreno in [33]. For comparison we also included the original $[1024, 524, 101]$ Goppa code used by McEliece.

System	Size of public key in bytes	Work factor	Information rate
Janwa & Moreno [171, 109, 61]	$5,914$	2^{61}	0.637
Srivastava [248, 160, 89]	$14,080$	2^{87}	0.645
Goppa [1024, 524, 101]	$32,750$	2^{71}	0.512
Goppa [2048, 1608, 81]	$88,440$	2^{109}	0.785
Goppa [2048, 1278, 141]	$123,008$	2^{120}	0.624

Table 1: Comparison of different AG codes

Here the work factor refers to the generalized information-set-decoding attack.

To compute the size of the public key we assumed that only the redundant part is stored; that is, the generator matrix G' is transformed, such that $G' = (I_k|A)$. Now only the $k\times(n-k)$ matrix A has to be stored.

7 Conclusion

Algebraic-geometric codes clearly have the potential to greatly improve the use of the McEliece cryptosystem.

The huge class of alternant codes can be decoded with low complexity $\mathcal{O}(n \log^2 n)$, and for other classes of AG codes good decoding algorithms exist as well.

More importantly, we were able to significantly reduce the code length n, resulting in public keys being smaller by a factor 6 to 9 compared with Goppa codes having $n = 2048$. This makes the McEliece cryptosystem more interesting for real-world applications, and there is still much room for improvement.

However, one has to be careful. Although no known general attack is feasible for appropriate parameters, structural attacks (like the one against the McEliece cryptosystem using GRS codes) can greatly reduce the work factor needed to decypher a message. Unfortunately, it is still hardly possible to foresee such structural weaknesses of a given code. To smooth out this problem, further research is necessary.

References

[1] Adams, C.M. ; Meijer, H.: Security-related comments regarding McEliece public-key cryptosystem. In: *IEEE Trans. Inform. Theory* 35 (1989), Nr. 2, S. 454–455

[2] Arimoto, S.: Encoding and decoding of p-ary group codes and the correction system. In: *Information Processing in Japan* 2 (1961), S. 320–325

[3] Berlekamp, E. ; McEliece, R. ; Tilborg, H. van: On the inherent intractability of certain coding problems. In: *IEEE Trans. Inform. Theory* 24 (1978), Nr. 3, S. 384–386

[4] Berlekamp, E.R.: *Algebraic coding theory.* McGraw-Hill, New York, 1968

[5] Berson, T.A.: Failure of the McEliece Public-Key Cryptosystem Under Message-Resend and Related-Message Attack. In: *Submitted to Crypto 97*, 1997

[6] Beth, Th. ; Frisch, M. ; (Eds.), G.J. S.: *Public-Key Cryptography: State of the Art and Future Directions.* Springer-Verlag, 1992

[7] Blahut, R.E.: *Algebraic coding theory in one and two dimensions.* 1994. – Lectures at the Eindhoven Univ. Techn.

[8] Blake, I. ; Heegard, C. ; Høholdt, T. ; Wei, V.: Algebraic-Geometry Codes. In: *IEEE Trans. Inform. Theory* 44 (1998), Nr. 6

[9] Bras-Amorós, M. ; O'Sullivan, M.E. *The Correction Capability of the Berlekamp-Massey-Sakata Algorithm with Majority Voting.* 2004

[10] Canteaut, A. ; Sendrier, N.: Cryptoanalysis of the Original McEliece Cryptosystem. In: *In Proc. of ASIACRYPT '98*, 1998, S. 187–199

[11] Chabaud, F.: On the security of some cryptosystems based on error-correcting codes. In: *EUROCRYPT*, 1994, S. 131–139

[12] Dahl, C.: Fast decoding of codes from algebraic curves. In: *IEEE Trans. Inform. Theory* 40 (1994), S. 223–230

[13] Davies, D.W. ; Price, W.L.: *Security for computer networks.* John Wiley & Sons, 1984

[14] Driencourt, Y.: Some properties of elliptic codes over a field of characteristic 2. In: *Proceedings AAECC-3, Grenoble 1985, Lect. Notes Comp. Sc.* Bd. 229, 1986, S. 185–193

[15] Duursma, I.M.: *Decoding codes from curves and cyclic codes*, Eindhoven Univ. Techn., Diss., 1993

[16] Duursma, I.M.: Majority coset decoding. In: *IEEE Trans. Inform. Theory* IT-39 (1993), S. 1067–1071

[17] (ed.), G. J. S.: *Contemporary Cryptology: The science of Information Integrity.* IEEE Press, New Jersey, 1992

[18] Ehrhard, D.: *Über das Dekodieren Algebraisch-Geometrischer Codes*, Universität Düsseldorf, Diss., 1991

[19] Ehrhard, D.: Decoding algebraic-geometric codes by solving a key equation. In: *Proceedings AGCT-3, Luminy 1991, Lect. Notes Math.* Bd. 1518, 1992, S. 18–25

[20] Ehrhard, D.: Achieving the Designed Error Capacity in Decoding Algebraic-Geometric Codes. In: *IEEE Trans. Inform. Theory* 39 (1993), S. 743–751

[21] Engelbert, D. ; Overbeck, R. ; Schmidt, A. *A summary on the development of the McEliece Cryptosystem.* 2005

[22] Feng, G.-L. ; Rao, T.R.N.: Decoding of algebraic geometric codes up to the designed minimum distance. In: *IEEE Trans. Inform. Theory* IT-39 (1993), S. 37–45

[23] Feng, G.-L. ; Rao, T.R.N.: A simple approach for construction of algebraic-geometric codes from affine plane curves. In: *IEEE Trans. Inform. Theory* IT-40 (1994), S. 1003–1012

[24] Feng, G.-L. ; Tzeng, K.K.: A new procedure for decoding cyclic and BCH codes up to actual minimum distance. In: *IEEE Trans. Inform. Theory* IT-40 (1994), S. 1364–1374

[25] Gabidulin, E.M.: On Public-Key Cryptosystems Based on Linear Codes: Efficiency and Weakness. In: *Codes and Ciphers, Proc. 4th IMA Conference on Cryptography and Coding*, 1995

[26] Gibbon, J.K.: Equivalent codes and trapdoors to McEliece's public-key cryptosystem. In: *EUROCRYPT '91, Lect. Notes in CS*, 1991, S. 68–70

[27] Goppa, V.D.: Algebraic-Geometric Codes. In: *Math. USRR-Izv.* 21(1) (1983), S. 75–93

[28] Havemose, A.: *Decoding algebraic geometric codes*, Danmarks Tekniske Højskole, Diss., 1989

[29] Heiman, R. *On the security of cryptosystems based on linear error-correcting codes.* 1987

[30] Helgert, H.J.: Srivastava Codes. In: *IEEE Trans. Inform. Theory* IT-18 (1972), S. 292–297

[31] Høholdt, T. ; Pellikaan, R.: On the decoding of algebraic-geometric codes. In: *IEEE Trans. Inform. Theory* IT-41 (1995), S. 1589–1614

[32] Imai, H. ; Kobara, K.: On the one-wayness against chosen-plaintext attacks of the Loidreau's modified McEliece PKC. In: *IEEE Trans. Inform. Theory* 49 (2003), Nr. 12, S. 3160–3168

[33] Janwa, H. ; Moreno, O.: *McEliece Public Key Cryptosystem Using Algebraic-Geometric Codes.* Boston : 1996 Kluwer Academic Publishers, 1996, S. 293–307

[34] Jensen, H. E. ; Nielsen, R.R. ; Høholdt, T.: Performance Analysis of a Decoding Algorithm for Algebraic-Geometry Codes. In: *IEEE Trans. Inform. Theory* 45 (1999), S. 1712–1717

[35] Jorissen, F.: A security evaluation of the public-key cipher system proposed by McEliece, used as a combined scheme. In: *ESAT report K.U.Leuven* (1986)

[36] Justesen, J. ; Larsen, K.J. ; Elbrønd Jensen, H. ; Havemose, A. ; Høholdt, T.: Construction and decoding of a class of algebraic geometric codes. In: *IEEE Trans. Inform. Theory* IT-35 (1989), S. 811–821

[37] Justesen, J. ; Larsen, K.J. ; Jensen, H. E. ; Høholdt, T.: Fast decoding of codes from algebraic plane curves. In: *IEEE Trans. Inform. Theory* IT-38 (1992), S. 111–119

[38] Kirfel, C. ; Pellikaan, R.: The minimum distance of codes in an array coming from telescopic semigroups. In: *IEEE Trans. Inform. Theory* IT-41 (1995), S. 1720–1732

[39] Kobara, K. ; Imai, H.: Countermeasure against Reaction Attacks (in Japanese). In: *The 2000 Symposium on Cryptography and Information Security*, 2000

[40] Kobara, K. ; Imai, H.: Semantically Secure McEliece Public-Key Cryptosystems - Conversions for McEliece PKC. In: *In Proc. of 4th International Workshop on Practice and Theory in Public Key Cryptosystems (PKC '01)*, 2001, S. 19–35

[41] Korzhik, V.I. ; Turkin, A.I.: Cryptanalysis of McEliece's public-key cryptosystem. In: *EUROCRYPT '91, Lect. Notes in CS*, 1991, S. 68–70

[42] Lee, P.J. ; Brickell, E.F.: An observation on the security of McEliece's public-key cryptosystem. In: *EUROCRYPT '88, Lect. Notes in CS*, 1988, S. 275–280

[43] Li, Y.X. ; Deng, R.H. ; Wang, X.M.: The equivalence of McEliece's and Niederreiter's public-key cryptosystems. In: *IEEE Trans. Inform. Theory* 40 (1994), S. 271–273

[44] Loidreau, P.: Strengthening McEliece Cryptosystem. In: *In Proc. of ASIACRYPT 2000*, 2000

[45] Loidreau, P. ; Sendrier, N.: Weak Keys in the McEliece Public-Key Cryptosystem. In: *IEEE Trans. Inform. Theory* 47 (2001), Nr. 3, S. 1207–1211

[46] MacWilliams, F.J. ; Sloane, N.J.A.: *The Theory of Error Correcting Codes.* North-Holland, 1977. – ISBN 0–444–85193–3

[47] Madelung, Y.: Implementation of a decoding algorithm for AG-codes from the Hermitian curve. In: *report IT-93-137* (1993)

[48] Massey, J.L.: Shift-register synthesis and BCH decoding. In: *IEEE Trans. Inform. Theory* IT-15 (1969), S. 122–127

[49] McEliece, R.J.: A Public-key cryptosystem based on algebraic coding theory. In: *DNS Progress Report* (1978), S. 114–116

[50] Menzes, A.J. ; Oorschot, P.C. ; Vanstone, S.A.: *McEliece public-key encryption.* CRC Press, 1997, S. 299

[51] O'Sullivan, M.E.: Decoding of codes defined by a single point on a curve. In: *IEEE Trans. Inform. Theory* 41 (1995), S. 1709–1719

[52] O'Sullivan, M.E. *A Generalization of the Berlekamp-Massey-Sakata Algorithm.* 2001

[53] Park, C.S.: Improving code rate of McEliece's public-key cryptosystem. In: *Electronic letters* 25 (1989), S. 1466–1467

[54] Pellikaan, R. ; Shen, B.-Z. ; Wee, G.J.M. van: Which linear codes are algebraic-geometric? In: *IEEE Trans. Inform. Theory* IT-37 (1991), S. 583–602

[55] Peterson, W.W.: Encoding and error-correction procedures for the Bose-Chauduri codes. In: *IRE Trans. Inform. Theory* IT-6 (1960), S. 459–470

[56] Porter, S.C.: *Decoding codes arising from Goppa's construction on algebraic curves*, Yale Univ., Diss., 1988

[57] Porter, S.C. ; Shen, B.-Z. ; Pellikaan, R.: On decoding geometric Goppa codes using an extra place. In: *IEEE Trans. Inform. Theory* IT-38 (1992), S. 1663–1676

[58] Rao, T.R.N. ; Nam, K.-H.: Private-key algebraic-code encryption. In: *IEEE Trans. Inform. Theory* IT-35 (1989), Nr. 4, S. 829–833

[59] Sakata, S.: Finding a minimal set of linear recurring relations capable of generating a given finite two-dimensional array. In: *Journal of Symbolic Computation* 5 (1988), S. 321–337

[60] Sakata, S.: Extension of the Berlekamp-Massey algorithm to N dimensions. In: *Journal of Symbolic Computation* 84 (1990), S. 207–239

[61] Sakata, S. ; Jensen, H. E. ; Høholdt, T.: Generalized Berlekamp-Massey decoding of algebraic geometric codes up to half the Feng-Rao bound. In: *IEEE Trans. Inform. Theory* 41, Nr. 6

[62] Sakata, S. ; Justesen, J. ; Madelung, Y. ; Jensen, H. E. ; Høholdt, T.: Fast decoding of algebraic geometric codes up to the designed minimum distance. In: *IEEE Trans. Inform. Theory* 41, Nr. 6

[63] Sakata, S. ; Justesen, J. ; Madelung, Y. ; Jensen, H. E. ; Høholdt, T.: A fast decoding method of AG codes from Miura-Kamiya curves C_{ab} up to Half the Feng-Rao bound. In: *Finite Fields and their Applications* 11 (1995), S. 83–101

[64] Sakata, S. ; Leonard, D.A. ; Jensen, H. E. ; Høholdt, T.: Fast Erasure-and-Error Decoding of Algebraic Geometry Codes up to the Feng-Rao Bound. In: *IEEE Trans. Inform. Theory* 44 (1998), Nr. 4, S. 1558–1564

[65] Sarwate, D.V.: On the Complexity of Decoding Goppa Codes. In: *IEEE Trans. Inform. Theory* 23 (1977), S. 515–516

[66] Sendrier, N.: Finding the permutation between equivalent linear codes: the support splitting algorithm. In: *IEEE Trans. Inform. Theory* 46 (2000), S. 1193–1203

[67] Serre, J.-P.: *Nombres de points des courbes Algébriques sur F_q.* 1983. – Séminaire de Théorie des Nombres de Bordeaux

[68] Serre, J.-P.: *Rational points on curves over finite fields, "q large"*. 1985. – Lectures given at Harvard University, Notes by F. Gouv́ea

[69] Shen, B.-Z.: *Algebraic-geometric codes and their decoding algorithm*, Eindhoven Univ. of Techn., Diss., 1992

[70] Shen, B.-Z.: On encoding and decoding of the codes from Hermitian curves. In: *Cryptography and Coding III, the IMA Conference Proceedings Series* New Series Number 45 (1993), S. 337–356

[71] Sidelnikov, V.M. ; Shestakov, S.O.: On the Insecurity of Cryptosystems Based on Generalized Reed-Solomon Codes. In: *Discrete Math.* 1 (1992), Nr. 4, S. 439–444

[72] Skorobogatov, A.N. ; Vlăduţ, S.G.: On the decoding of algebraic-geometric codes. In: *IEEE Trans. Inform. Theory* IT-36 (1990), S. 1051–1060

[73] Stern, J.: A method for finding codewords of small weight. In: *In Proc. of Coding Theory and Applications*, 1989, S. 106–113

[74] Stichtenoth, H.: *Algebraic Function Fields and Codes.* Springer-Verlag, 1993. – ISBN 3–540–56489–6

[75] Stoll, M.: *Algebraische Kurven.* 2001. – Vorlesung im Wintersemester 2001/2002

[76] Sugiyama, Y. ; Kasahara, M. ; Hirasawa, S. ; Namekawa, T.: A method for solving key equation for decoding Goppa codes. In: *Information and Control* 27 (1975), S. 87–99

[77] Sugiyama, Y. ; Kasahara, M. ; Hirasawa, S. ; Namekawa, T.: Some Efficient Binary Codes Constructed Using Srivastava Codes. In: *IEEE Trans. Inform. Theory* 21 (1975), S. 581–582

[78] Tilburg, J. van: On the McEliece public-key cryptosystem. In: *CRYPT '88, Lect. Notes in CS*, 1988

[79] Voss, C. ; Høholdt, T.: A family of Kummer extensions of the Hermitian function field. In: *Communications in Algebra* 23 (1995), Nr. 4, S. 1551–1567