McEliece Cryptosystems

Constructed on Algebraic Geometry Codes

Martin Sig Nørbjerg

P7 Project, Group 5.239a, Mathematics

Dept. of Mathematical Sciences
Skjernvej 4A
DK-9220 Aalborg Ø
http://math.aau.dk

Title
McEliece Cryptosystems

Themes
Algebraic Geometry Codes
Reimann-Roch Spaces
Post Quantum Cryptography

Project Period
Autum Semester 2023

Project Group
Group 5.239a

Participants
Martin Sig Nørbjerg

Supervisor
Matteo Bonini

Page Numbers
44

Date of Completion
November 13, 2023

Abstract

In this bachelors project we study the theory of algebraic geometry codes, these are linear error correcting codes constructed using the theory of algebraic geometry and in particular the theory of algebraic curves. Using the Reimann Roch Theorem we show that there exists good lower bounds on the parameters of these codes. Finally, we discuss the asymptotic properties of these codes and in particular how one can construct algebraic geometry codes which execed the asymptotic Giblert-Varshamov bound.

Preface

This is a P7 project written at the Department of Mathematical Sciences at Aalborg University. As such it is expected to have the general knowledge acquired during a bachelors degree in mathematics. In addition it is expected that the reader is familiar with the following:

(i) The basics algebraic geometry, in particular the theory of protective regular and absolutely irreducible curves over $\mathbb{F}_q$

(ii) The basics of error correcting codes.

(iii) The basic construction and properties of algebraic geometry codes.

The work on the project took place from the 1. of September to the 20. of December 2023.

Sources are stated at the start of each chapter, or section if the sources used in the section differ from the sources, that are generally used within the chapter.

Definitions, algorithms, theorems, propositions, lemmas, corollaries, examples, and remarks are numbered according to each chapter and consecutively. Equations are also numbered according to each chapter but separately. The conclusions of proofs and examples are marked with ■ and □ respectively.

A table of notation and shorthands is given after the list of contents. Please note that some symbols may be used differently in different chapters.

Aalborg University, November 13, 2023

Martin Sig Nørbjerg
mnarbj20@student.aau.dk

Contents

Notation and Shorthands

Abstract Algebra

$\mathbb{F}_q$	The finite field with q elements.
$R[X_1, X_2, \ldots, X_n]$	The multivariate polynomial ring over R.
$X^{(k)}$	The mononomial $\prod_{i=1}^{n} X_i^{k_i}$.
R^*	The set of units in R.
$\langle S \rangle$	The ideal generated by the elements in the set S.
$\mathbb{K}/\mathbb{F}$	A field extension, meaning $\mathbb{F}$ is a subfield of the field $\mathbb{K}$.
$\overline{\mathbb{F}}$	An algebraic closure of the field $\mathbb{F}$.
$Rad(I)$	The radical of the ideal I.
$\mathrm{Ev}_{\mathcal{P}}$	The evaluation map.

Algebraic Geometry

$\Bbbk$	An arbitrary algebraically closed field.
$\mathbb{A}^n, \mathbb{P}^n$	The n-dimensional affine and projective space.
$V(S), V_{\mathbb{P}}(S)$	The affine and projective zero sets a set of polynomials S.
$I(X), I_{\mathbb{P}}(X)$	The affine and projective vanishing ideals a set of points X
$\mathcal{T}_Z$	The Zariski topology.
F^*, F_*	The homogenisation and dehomogenisation of the polynomial F.
$\mathcal{X}$	An affine or projective variety.
$\Bbbk[\mathcal{X}], \Bbbk(\mathcal{X})$	The coordinate ring and function field of the affine variety $\mathcal{X}$.
$\Bbbk_{\mathbb{P}}[\mathcal{X}], \Bbbk_{\mathbb{P}}(\mathcal{X}), \Bbbk[\mathcal{X}]$	The homogeneous coordinate ring and homogeneous function field and function field
$\Bbbk(\mathcal{X})$	The function field over $\mathcal{X}$.
$\mathcal{O}_P(\mathcal{X}), \mathfrak{m}_P$	The local ring of $\mathcal{X}$ at P, and it's maximal ideal.
v_P	The discrete valuation on $\mathcal{O}_P(\mathcal{X})$.
$I(P, \mathcal{X}, \mathcal{Y})$	The intersection multiplicity of $\mathcal{X}$ and $\mathcal{Y}$ at the point p.
$Div(\mathcal{X})$	The set of divisors on $\mathcal{X}$.
$\mathrm{supp}(D)$	The support of a divisor D.
(f)	The principal divisor of $f \in \Bbbk[\mathcal{X}]$.
$L(D), \ell(D)$	A special vector space of divisors and it's dimension.

Coding theory

$\mathcal{C}$	A linear code.
G	A generator matrix.
H	A parity check matrix.
d	The Hamming metric.
wt	The Hamming weight.
$\overline{B_r}(x)$	The hamming ball of radius r with center in x.
H_q	The q-ary entropy function.
$\mathcal{C}^\perp$	The dual code of $\mathcal{C}$.
δ	The relative distance.
R	The relative distance.

Algebraic Geometry Codes

$\mathcal{C}_{D,G}$	A Goppa Code.
ψ	The extended Frobenius map
$N_q(\mathcal{X})$	The number of $\mathbb{F}_q$-rational points on $\mathcal{X}$
$N_q^*(g)$	The maximum number of rational points on a curve of genus g

1 Introduction

Most of the widely used public key cryptosystems are build upon problems in abstract algebra and number theory, that seem hard to solve on classical computers, notable examples include the following problems:

Problem 1.1 (Integer Factorization)**.** Given $n \in \mathbb{N}$, compute the prime factors of n.

Problem 1.2 (Discrete Logarithm)**.** Let $n \in \mathbb{N}$ and a be an element of a group $(G, \circ)$, then given $a^n = \underbrace{a \circ a \circ \cdots \circ a}_{n \text{ times}}$, compute n.

Problem 1.1 underpin the security of the well know RSA cryptostyem, see Lauritzen (2003)[Section 1.9], while Problem 1.2 underpin more recent cryptosystem, such as the Diffie-Hellman key exchange system[1], see Koblitz (1994)[Section 4.3]. However algorithms for solving these problems, in polynomial time, on a sufficiently powerfull quantum computer have are already well known. The previously mentioned algorithms where initially described by Peter Williston Shor in 1996, see Shor (1996). Instead the goal of this project will be to study two public key cryptosystems based on the following problems:

Problem 1.3 (General Decoding Problem)**.** Let $\mathcal{C} \subseteq \mathbb{F}_q^n$ be a linear code, let $t \in \mathbb{N}$ such that $t \leq n$ and $y \in \mathbb{F}_q^n$. Decide if there exists a codeword $c \in \mathcal{C}$, such that $d(c, y) \leq t$

Problem 1.4 (Coset Weight Problem)**.** Let $\mathcal{C} \subseteq \mathbb{F}_q^n$ be a linear code with parity check matrix H, $t \in \mathbb{N}$ such that $t \leq n$ and $He \in \mathbb{F}_q^{(n-k)}$. Then find $e \in \mathbb{F}_q^n$ such that $\text{wt}(e) \leq t$.

Both of the problems has been proved to be **NP**-complete, confer Berlekamp et al. (1978), for the case where $q = 2$. For a brief and informal introduction to the topic of **NP**-completeness we refer the reader to Appendix A.

The public key cryptosystem based on Problem **??** is named after Robert J. McElice, who originally proposed it in 1978. While the cryptosystem based on Problem 1.4 is named after Harald Niederreiter who first proposed it in 1986.

TODO Skriv en oversigt over layouted

[1] This scheme actually relies on the fact that given g^n and g^m it seems to be computationally infeasible to compute g^{nm}, however if there exists a method for solving Problem 1.2, then this problem is solved trivially.

2 Code Based Cryptography

The following introduction of the McElice public key cryptosystem (abreviated McElice PKCS), is based upon Singh (2020). Before we can introduce the McElice PKCS, we will need to introduce some basic concepts from error correcting codes.

Definition 2.1. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be $[n,k]_q$ codes, with generator matrices G_1 and G_2 respectively then:

(i) If there exists a permutation matrix $P \in \mathbb{F}_q^{k\times k}$ such that $G_1 = G_2P$, then $\mathcal{C}_1$ and $\mathcal{C}_2$ are called *permutation equivalent*.

(ii) The codes $\mathcal{C}_1$ and $\mathcal{C}_2$ are called *equivalent*, denoted $\mathcal{C}_1 \sim \mathcal{C}_2$, if there exists a matrix $S \in GL_n(\mathbb{F}_q)$ and permutation matrix $P \in \mathbb{F}_q^{n\times n}$ such that $G_1 = SG_2P$.

Remark 2.2. The relation $\sim$, is indeed an equivalence relation:

(i) If G is a generator matrix, then $G = G$ so $\sim$ is reflective.

(ii) Additionally if $G_1 = SG_2P$ then $S^{-1}G_1P^{-1} = G_2$ so $\sim$ is symmetric.

(iii) Finally, to show that $\sim$ is transitive, let $G_1 = S_1G_2P_1$ and $G_2 = S_2G_3P_2$. Then $G_1 = (S_1S_2)G_3(P_2P_1)$.

Two equivalent codes, clearly share the same dimension, after all their generator matricies have the same number of rows. However it is not immediately obvious that they share the same minimum distance.

Proposition 2.3. *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be equivalent $[n,k]_q$ codes, with generator matricies G_1 and G_2 respectively, such that $G_1 = SG_2P$, for some $S \in GL_n(\mathbb{F}_q)$ and permutation matrix $P \in \mathbb{F}_q^{k\times k}$: Then:*

(i) *If $H_2 \in \mathbb{F}_q^{n\times k}$ is a parity check matrix of $\mathcal{C}_2$ then the matrix H_2P is a parity check matrix of $\mathcal{C}_1$.*

(ii) *Additionally $d(\mathcal{C}_1) = d(\mathcal{C}_2)$.*

Proof. We start by proving Assertion (i). We do this by showing that $(H_2P)G_1^T = 0_{k\times k}$ meaning $\mathcal{C}_1 \subseteq \text{null}(H_2P)$ and that $\dim_{\mathbb{F}_q}(\text{null}(H_2P)) = k$. The fact that $\dim_{\mathbb{F}_q}(\text{null}(H_2P)) = k$ follows from the fact that P is a permutation matrix and hence:

$$\dim_{\mathbb{F}_q}(\text{null}(H_2P)) = \dim_{\mathbb{F}_q}(\text{null}(H_2))\dim_{\mathbb{F}_q}(\mathcal{C}_\in) = k$$

Next using the fact that $G_1 = SG_2P$ we see that:

$$H_2PG_1^T = H_2P(P^TG_2^TS^T) \overset{(a)}{=} H_2G_2^TS^T \overset{(b)}{=} 0_{n\times n}$$

where equality (a) follows as P is a permutation matrix and hence orthogonal, and equality (b) as $H_2G_2^T = 0_{n\times n}$.

Continuing Assertion (ii) follows by combining Assertion (i) with the fact that the minimum distance of a code with parity check matrix H, corresponds with the minimum number of linearly dependent columns of H. Finally we conclude the proof by noting that H_2P has the same columns as H_2. ■

Definition 2.4. Let $\mathcal{C}$ be a $[n,k,d]_q$ code and let $t \leq \left\lfloor \frac{d-1}{2} \right\rfloor$. A *$t$-error correcting decoder* for $\mathcal{C}$ is a mapping $Dec_{\mathcal{C}} : \mathbb{F}_q^n \to \mathcal{C} \cup \{?\}$ which satisfies the condition that $Dec_{\mathcal{C}}(y) = c$ whenever $y = c + e$, with $c \in \mathcal{C}$ and $e \in \mathbb{F}_q^n$ such that $\mathrm{wt}(e) \leq t$, and $Dec_{\mathcal{C}}(y) = ?$ otherwise. An algorithm which implements a t-error correcting decoder for $\mathcal{C}$ is called a *t-error correcting decoding algorithm* for $\mathcal{C}$.

Remark 2.5. We often simply refer to the t-error correcting decoder $Dec_{\mathcal{C}}$ as a *decoder* and any algorithm which implements a decoder for $\mathcal{C}$ as an *decoding algorithm* for $\mathcal{C}$.

We have now covered all of the necessary tools needed to introduce the McElice PKCS: Let $\mathcal{C}$ be an $[n,k,d]_q$ code with generator matrix G and an efficient t-error correcting decoding algorithm $Dec_{\mathcal{C}}$, furthermore let $S \in GL_{k\times k}(\mathbb{F}_q)$ and $P \in \mathbb{F}_q^{n\times n}$ be a random permutation matrix. An overview of the McElice PKCS is given below in Algorithm 2.6.

Algorithm 2.6 McElice PKC

private key $(Dec_{\mathcal{C}}, S, P)$
public key $(G' = S \cdot G \cdot P, t)$

procedure McElice Encryption(m: plain message)
  $c \leftarrow m^TG'$
  Choose $e \in \mathbb{F}_q^n$ uniformly, such that $\mathrm{wt}(e) = t$.
  return $c + e$

procedure McElice Decryption(y: received chipher-text)
  $y' \leftarrow yP^{-1}$
  $m' \leftarrow Dec_{\mathcal{C}}(y')$
  return $m'S^{-1}$

Proposition 2.7. *Given some $y = m^TG' + e$, with $e \in \mathbb{F}_q^n$ such that $\mathrm{wt}(e) = t$, the decryption algorithm described in Algorithm 2.6 yiels the correct message m^T.*

Proof. Sticking to the notation used in Algorithm 2.6 we have:

$$y' := (m^TG' + e)P^{-1} = m^TSG + eP^{-1}$$

However as $P^{-1} = P^T$ is also permutation matrix, we see that $\mathrm{wt}(eP^{-1}) = t$. Hence we may apply our t-error correcting decoder $Dec_{\mathcal{C}}$ to y' and get $m' := m^TS$ now multiplying m' by S^{-1} we obtain m^T. ■

Remark 2.8. The matrix G' will be the generator matrix of another $[n,k]_q$ code $\mathcal{C}'$. By Proposition 2.3 $d(\mathcal{C}') = d(\mathcal{C})$. Hence the decoding of $m^TG' + e$ also makes sense. In fact the decoding $m^TG' + e$ is one of the ways to attack the McElice PKCS. We will discuss this in more detail in Sections 2.1 and 2.4.

We mentioned earlier that we would study public key cryptosystem which where bases on the general decoding problem, See Problem 1.3. However the McElice PKCS is actually based on the following "weaker" problem, in the sense that an efficient algorithm for solving this problem doesn't necessarily yield an efficient solution to Problem 1.3. However if $\mathbf{NP} = \mathbf{P}$, meaning that Problem 1.3 could be solved in polynomial time, then the McElice PKCS would be venerable to attack.

Problem 2.9 (McElice Problem)**.** Given (G', t) and a ciphertext y find the unique $m \in \mathbb{F}_q^k$ such that $\mathrm{wt}(m^TG' - c) = t$.

The main difference between Problem 2.9 and Problem 1.3 is that we are provided with a basis of $\mathcal{C}'$ in Problem 2.9, since we know G'. However the idea is to have as little structure of the underlying code revealed by G' as possible, to make the problem closer to Problem 1.3.

In general we have two kinds of attacks on the McElice PKCS

(i) *Structural Attacks*: Where we try to extract information about the underlying code, using G'. If we obtain enough information we may be able to implement our own efficient t-error correcting decoding algorithm.

(ii) *Generic Attacks*: Where we try to construct efficient t-error correcting decoding algorithms given (G', t), without concerning our selves with the underlying structure of $\mathcal{C}$, we will refer to such decoding algorithms as being *generic*.

Two of the main goals of this project is to investigate these attacks. Understanding of structural attacks allows us to gain insights into which families of codes constitutes "good" candidates for use in the McElice PKCS. While generic attacks allows us to measure the security of the McElice PKCS constructed on the codes resilient to structural attacks.

2.1 Syndrome Decoding

The following section is based on Huffman and Pless (2003)[Subsection 1.11.2]. We start by introducing a way to partition $\mathbb{F}_q^n$ using a linear code.

Definition 2.10. Let $\mathcal{C}$ be a $[n,k]_q$ code and let $y \in \mathbb{F}_q^n$, then the *coset* of $\mathcal{C}$ determined by y is defined as:

$$\mathcal{C} + y := \{c + y | c \in \mathcal{C}\} =: y + \mathcal{C}$$

We note that these cosets are nothing more, than the cosets found in group theory, as $(\mathcal{C}, +)$ forms a subgroup of the abelian group $(\mathbb{F}_q^n, +)$. Hence the results of the following proposition should come as no suprice, never the less, we will provide a proof.

Proposition 2.11. *Let $\mathcal{C}$ be a $[n,k]_q$ code, then the following holds for all $y, y' \in \mathbb{F}_q^n$:*

(i) *If* $y' \in \mathcal{C} + y$ *then* $\mathcal{C} + y = \mathcal{C} + y'$.

(ii) *If* $\mathcal{C} + y \neq \mathcal{C} + y'$ *then* $(\mathcal{C} + y) \cap (\mathcal{C} + y') = \emptyset$.

(iii) *There are* q^{n-k} *disjoint cosets of* $\mathcal{C}$.

Proof. We start with Assertion (i): If $y' \in \mathcal{C} + y$, then $y' = c + y$ for some $c \in \mathcal{C}$, hence $y' + (-c) = y$. Now since $\mathcal{C}$ is a vector space we see that

$$\underbrace{y' + (-c)}_{=y} + c' \in \mathcal{C} + y \text{ for all } c' \in \mathcal{C}$$

so $y' + \mathcal{C} \subseteq y + \mathcal{C}$. We may apply an similar argument to get the other inclusion, thus $\mathcal{C} + y = \mathcal{C} + y'$.

Continuing with Assertion (ii): Assume for the sake of contradiction that $\mathcal{C} + y \neq \mathcal{C} + y'$ and $(\mathcal{C} + y) \cap (\mathcal{C} + y') \neq \emptyset$, then pick $x \in (\mathcal{C} + y) \cap (\mathcal{C} + y')$. By Assertion (i) we have $\mathcal{C} + y = \mathcal{C} + x = \mathcal{C} + y'$ which is clearly a contradiction.

Finally Assertion (iii) follows by the Lagrange's index theorem as:

$$q^n = \left|\mathbb{F}_q^n\right| = \left|\mathbb{F}_q^n/\mathcal{C}\right| |\mathcal{C}| = \left|\mathbb{F}_q^n/\mathcal{C}\right| q^k$$

implies that $\left|\mathbb{F}_q^n/\mathcal{C}\right| = q^{n-k}$. The fact that the cosets are disjoint follows directly from Assertion (ii). ■

Definition 2.12. Let $\mathcal{C}$ be a $[n,k]_q$ code with parity check matrix $H \in \mathbb{F}_q^{(n-k)\times n}$, the syndrome map:

$$S_H : \mathbb{F}_q^n \ni y \mapsto Hy \in \mathbb{F}_q^{(n-k)}$$

Additionally for any $y \in \mathbb{F}_q^n$ we say that $S_H(y)$ is the *syndrome* associated with y.

Remark 2.13. Clearly $S_H(y) = 0$ if and only if $y \in \mathcal{C}$, after all this is exactly the definition of a parity check matrix. In addition we have $S_H(y + y') = S_H(y) + S_H(y')$ since matrix vector multiplication is distributive.

Lemma 2.14. *Let* $\mathcal{C}$ *be a* $[n,k]_q$ *code with parity check matrix* H *and* $y, y' \in \mathbb{F}_q^n$*, then* $S_H(y) = S_H(y')$ *if and only if* y *and* y' *are in the same coset of* $\mathcal{C}$.

Proof. Firstly we note that for all $y, y' \in \mathbb{F}_q^n$ there exists a $x \in \mathbb{F}_q^n$ such that $y = y' + x$ and hence $S_H(y) = S_H(y') + S_H(x)$ per Remark 2.13. Now if $S_H(y) = S_H(y')$ we must have $S_H(x) = 0$ meaning $x \in \mathcal{C}$, again by Remark 2.13. The other implication follows by a similar argument as y and y' being in the same coset, implies that there exists a codeword $c \in \mathcal{C}$ such that $y = y' + c$ and hence $S_H(y) = S_H(y') + S_H(c) = S_H(y')$ by Remark 2.13. ■

Definition 2.15. If $\mathcal{C}$ is a $[n,k]_q$ code, with parity check matrix $H \in \mathbb{F}_q^{(n-k)\times k}$, then we define the *syndrome lookup table* (SLT) $S_H^* : \mathbb{F}_q^n \to \mathbb{F}_q^n$ as

$$S_H^*(y) = \underset{e \in y+\mathcal{C}}{\arg\min}\, \mathrm{wt}(e)$$

Furthermore the vector $S_H^*(y)$ is called the *coset leader* of $y + \mathcal{C}$.

Next we introduce the syndrome decoding algorithm, which requires will require such a syndrome lookup table, alternatively we could when decoding a received word y go through the elements of $y+\mathcal{C}$ to find the coset leader, however this is impractical and we instead use a syndrome lookup table to avoid unecessary computation. Hence we will first introduce a procedure for constructing such a lookup table.

Algorithm 2.16 Syndrome Lookup Table Construction and Syndrome Decoding

procedure SLT CONSTRUCTION($\mathcal{C}$: a $[n,k]_q$ code, H: parity check matrix of $\mathcal{C}$)
  $S_H^* \leftarrow \emptyset$ ▷ We will view this as a mapping.
  for $i \in \{1, 2, \ldots, n\}$ **do**
    for $x \in \mathbb{F}_q^n$ with $\mathrm{wt}(x) = i$ **do**
      if $S_H(x)$ is not already in the codomain of S_H^* **then**
        $S_H^* \leftarrow S_H^* \cup \{(S_H(x), x)\}$
        if $|S_H^*| = q^{n-k}$ **then**
          return S_H^*

procedure SYNDROME DECODING(y: received word, H: a parity check matrix
  S_H^*: a syndrome lookup table)
  $\hat{e} \leftarrow S_H^*(S_H(y))$
  return $y - \hat{e}$

Remark 2.17. The syndrome lookup table S_H^* constructed in Algorithm 2.16, differs from the syndrome lookup table as defined in Definition 2.15, in that it takes the syndrome of y and returns the coset leader. However the function $S_H^* \circ S_H$ do confine to Definition 2.15.

Remark 2.18. Since $\hat{e}$ is the coset leader of $y+\mathcal{C}$, we know that $y-\hat{e}$ is our nearest neighbor estimate for c, since $\mathrm{wt}(\hat{e})$ is minimal.

We also note that since every coset contains a finite number of words, we may simply iterate through them to find the one which minimizes the hamming weight. By now the diligent reader, might start to question how does all this relate to solving the McEliece problem, since we simply know a generator matrix? This might seem like a problem, however it is pretty trivial to solve, we start by proving the following theorem:

Theorem 2.19. *Let $\mathcal{C}$ be a $[n,k]_q$ code and $A \in \mathbb{F}_q^{(n-k)\times k}$. Then $G = \begin{bmatrix} I_k & A \end{bmatrix}$ is a generator matrix for $\mathcal{C}$ if and only if $H = \begin{bmatrix} -A^T & I_{n-k} \end{bmatrix}$ is a parity check matrix for $\mathcal{C}$.*

Proof. We start by noting that $HG^T = -A^T + A^T = 0_{k\times k}$ hence all rows of G are in $\mathrm{null}(H)$. Hence if G is a generator matrix of $\mathcal{C}$, then H is a parity check matrix of $\mathcal{C}$. On the otherhand if H is a parity check matrix for $\mathcal{C}$, then G is a generator matrix for $\mathcal{C}$ since $\dim_{\mathbb{F}_q}(\mathrm{null}(H)) = n - \mathrm{rank}(H) = n - (n-k) = k$ and $\mathrm{rank}(G) = k$. Hence the rows of G must form a $\mathbb{F}_q$-basis of $\mathrm{null}(H) = \mathcal{C}$. ■

Hence if we receive a generator matrix for a $[n,k]_q$ code C, we can simply apply Gaussian elimination algorithm to get a generator matrix G which is in reduced echelon form, note that G isn't neccecarily standard form, but by multiplying our generator matrix G by a permutation matrix P, we may obtain a generator matrix G', which is in standard form, since G has k pivots as $\dim(\mathcal{C}) = k$. We note that G' is not nessecarily a generator matrix for $\mathcal{C}$ but rather a generator matrix for a code $\mathcal{C}'$ permutation equvilent to $\mathcal{C}$. Applying

Theorem 2.19 we obtain a parity check matrix H' for $\mathcal{C}'$, which we transform into a parity check matrix H of $\mathcal{C}$ by setting $H = P^{-1}H'$

Finally we consider the time and space complexity of Algorithm 2.16. We will consider the two procedures described seperately, since we only need to create one syndrome lookup table once, to decode a given code.

Since SLT Construction loops over each coset of $\mathcal{C}$, of which there is q^{n-k}, by 2.11(iii). To find the coset leader we can assume that we simply iterate over the words in $y + \mathcal{C}$, of which there is q^k, hence we see that SLT Construction has a time complexity of $O(q^n)$. In addition since each syndrome and coset leader pair needs to be stored, we see that SLT Construction has a space complexity of $O(q^{n-k})$.

Continuing with the Syndrome Decoding procedure, we once again get a space complexity of $O(q^{n-k})$, since we have to store S^*_H, while the time complexity depends on the underlying datastructure chosen to represent S^*_H, however it will be atleast $O((n-k)n)$ since we have to compute $S_H(y) = Hy$ and H is a $(n-k) \times n$ matrix

2.2 The Niederreiter Public Key Cryptosystem

Suppose $\mathcal{C}$ is a $[n, k, d]_q$ code with parity check matrix H and an efficient t-error correcting decoding algorithm $dec_{\mathcal{C}}$. Furthermore let $S \in GL_{(n-k)\times(n-k)}(\mathbb{F}_q)$ and $P \in \mathbb{F}_q^{n\times n}$ be a random permutation matrix. The Neiderreiter PCKS (based on $\mathcal{C}$), which allows for encryption of a plain message $e \in \mathbb{F}_q^n$ with $\mathrm{wt}(e) = t$, is described below in Algorithm 2.20.

Algorithm 2.20 The Neiderreiter PKCS

private key $(Dec_{\mathcal{C}}, S, P)$
public key $(H' = S \cdot H \cdot P, t)$

procedure Neiderreiter Encryption(e: plain message with $\mathrm{wt}(e) = t$)
  return $H'e$

procedure Neiderreiter Decryption(y: received ciphertext)
  Find $z \in \mathbb{F}_q^n$ such that $Hz = S^{-1}y$ using linear algebra.
  $c \leftarrow Dec_{\mathcal{C}}(z)$
  return cP^{-1}

Finding a $z \in \mathbb{F}_q^n$ such that $Hz = S^{-1}y$ is simply a matter of solving a linear system. Additionally since $S^{-1}y = S^{-1}SHPe = HPe$ we see that $c := Dec_{\mathcal{C}}(z)$ is the closest codeword to HPe so $e = z - cP^{-1}$.

2.2.1 The Equivalence Between the Neiderreiter and the McElice PCKS

Consider the $[n, k, d]_q$ code $\mathcal{C}$ with generator matrix G and parity check matrix H. We demonstrate that the McElice and Neither cryptosystems based on $\mathcal{C}$ have an equivalent level of security. That is if a there exists an efficient attack on the McElice PCKS (based on $\mathcal{C}$), then there exists an efficient attack on the Neither PCKS (based on $\mathcal{C}$).

Hence we let G' and H' be the public keys of the McElice and Neither PCKS respectively.

Suppose we have a message $m \in \mathbb{F}_q^k$, we encrypt the message a obtain $y \in \mathbb{F}_q^n$, using Algorithm 2.6. That is:

$$y = m^T G' + e$$

where $e \in \mathbb{F}_q^n$ with $\text{wt}(e) = t$. Given G' we may obtain a parity check matrix H' for the code generated confer Theorem 2.19. Multiplying by $(H')^T$ we get:

$$y(H')^T = mG'(H')^T + e(H')^T = e(H')^T \tag{2.1}$$

where the last equality follows as $G'(H')^T = 0$. Additionally since y and H' are public, the righthand side of Equation (2.1) can easily be computed. Furthermore since $\text{wt}(e) = t$, we see that we may compute the error e efficiently provided we have an efficient attack on the Neiderreiter PCKS. Hence an efficient attack on the Neiderreiter PCKS would lead to an efficient attack on the corresponding McElice PCKS with very little overhead.

Conversely assume that we have a message $e \in \mathbb{F}_q^n$ such that $\text{wt}(e) = t$. If we encrypt the message to obtain $y \in \mathbb{F}_q^{(n-k)}$, using Algorithm 2.20. That is:

$$y = H'e$$

Again we may obtain generator matrix G' of the code $\text{null}(H')$, since the null space of a matrix is invariant under row operations, thus we may apply Theorem 2.19. Using basic linear algebra one my find a vector $z \in \mathbb{F}_q^n$ with $\text{wt}(z) \geq t$ such that $y = H'z$ after all d is the minimum number of linearly independent columns of H' and $t \leq \lfloor \frac{d-1}{2} \rfloor$. Hence:

$$y = H'z \text{ and } z = yG' + e$$

Hence e could be extracted efficiently provided that there exists an efficient algorithm for breaking the McElice PCKS.

2.3 Advantages and Drawbacks of McElice and Neiderreiter

In this final subsection we will briefly discuss some of the advantages and drawbacks of using each system. Both compared to each other and compared to traditional public key cryptography systems. The results are summarized in Table 2.1. The encryption procedure of the McElice PCKS is very efficient additionally there exists no well known quantum algorithm for breaking the McElice PCKS. The primary con of the McElice PCKS is the large key size, for example the original proposal by McElice, used a $[1024, 524]_2$ classical Goppa code, see Definition 5.2. Due to the dimensions of the code the public key was roughly $524 \cdot 1024 = 536576$ bits or about 67.1 KB while yielding a security level of about 65 bits (remember to do the actual calculation). Meaning an attacker would have to perform 2^{65} operations to break the encryption.. In comparison using RSA a public key size of only 3072 bits is sufficient to yield a security level of 128 bits, refer Barker (2020)[Table 2].

For this reason much research has been done with the objective of lowering the public key size. However even though many of these proposals have succeeded in lowering the public key size, they often come with security issues.

The Neiderreiter PCKS has similar drawbacks as the McElice PCKS, in that the key size is very large compared to traditional public key cryptography systems, like RSA. However it has an additional drawback compared to the McElice PCKS, namely that the message has to have weight t while having length n. Where as the only restriction on the message in the

McElice PCKS is that it should have length n. On the otherhand one of the advantages of using the Neiderreiter PCKS is that it offers a smaller key size. Since the parity check matrix H' may be published in systematic form, that is $H' = [A|I_{(n-k)\times(n-k)}]$, while keeping the security level the same, we prove this below in Proposition 2.22. We will however first need a small lemma:

Lemma 2.21. *Let H be parity check matrix for the $[n,k,d]_q$ code $\mathcal{C}$, $t \leq \left\lfloor \frac{d-1}{2} \right\rfloor$ and $x, x' \in \mathbb{F}_q^n$. Then $\mathrm{wt}(x) = \mathrm{wt}(x') = t$ and $S_H(x) = S_H(x')$ implies $x = x'$.*

Proof. By Lemma 2.14 we have $x = x' + c$ for some $c \in \mathcal{C}$. Thus $c = x - x'$ which implies:

$$\mathrm{wt}(c) = \mathrm{wt}(x - x') \leq 2t < d$$

since $\mathrm{wt}(x) = \mathrm{wt}(x') = t$. Thus c must equal zero. ■

Proposition 2.22. *Let H be a parity check matrix for the $[n,k,d]_q$ code $\mathcal{C}$. Furthermore let $H' := UH$ be a systematic parity check matrix of $\mathcal{C}$, then any attack able to break the scheme using H' is able to break a scheme using H.*

Proof. Let $W_t = \left\{x \in \mathbb{F}_q^n | \mathrm{wt}(x) = t\right\}$ with $t \leq \left\lfloor \frac{d-1}{2} \right\rfloor$. Then S_H and $S_{H'}$, restricted to W_t, are injective by Lemma 2.21. Assume that ϕ breaks the system using H', that is $x = \phi(y) = S_{H'}^{-1}(y)$ for all $y \in S_{H'}(W_t)$.
Next suppose $y \in S_H(W_t)$, then $y = S_H(x) = Hx$ for some $x \in W_t$. Hence:

$$Uy = US_H(x) = UHx = S_{H'}(x) \in S_{H'}(W_t)$$

and $\phi(Uy) = x$. That is ϕ can be used to break the system using the parity check matrix H, we note that U can be obtained by performing elementary row operations on H. ■

Since H is allowed to be in standard form, may simply publish the first k columns of H assuming H is in systematic form, allowing for a much smaller public key size at least when k is relatively large compared to n. As an example consider McElices original proposal which used a $[1024, 524]_2$ classical Goppa code, meaning the public key of the equivalent neither system, could be published using $(1024 - 524) \cdot 524 = 26200$ or about 32.8 KB, which is about half the size of the public key in the equivalent McElice PCKS. Finally the contents of the discussion is summarized in Table 2.1.

System	Advantages	Drawbacks
McElice	Fast encryption No known efficient quantum attacks	Large key size (versus e.g. RSA)
Neiderreiter	Smaller key size (versus McElice) No known efficient quantum attacks	Large key size (versus e.g. RSA) Messages must be of weight t

Table 2.1: *Advantages and drawbacks of the McElice and Neiderreiter PCKS.*

2.4 Information Sets

The following section will be based on Peters (2010)[Sections 3-6], Lee and Brickell (1988) and Löndahl (2014)[Chapter 3]. We start by introducing some basic notation. Let $I \subseteq \{1, \ldots, n\}$ be non-empty. We will let G_I denote the matrix which consists of the columns of G such

that the ith column is in G_I if and only if $i \in I$ and we will similarly let y_I denote the restriction of $y \in \mathbb{F}_q^n$ to the coordinates indexed by I. Following the convention of Löndahl (2014) we let $\text{supp}(y) = \{i \in \{1, \dots, n\} \,|y_i \neq 0\}$.

Proposition 2.23. *Let $G \in \mathbb{F}_q^{k\times n}$ be the generator matrix of some code $\mathcal{C}$, and $A \in GL_k(\mathbb{F}_q)$, then $A^{-1}G$ is a generator matrix of $\mathcal{C}$ as well.*

Proof. Suppose $c \in \mathcal{C}$ then $c = m^T G$ for some $m \in \mathbb{F}_q^k$, in addition we may solve the equation $x^T A^{-1} = m^T$ for $x \in \mathbb{F}_q^k$. Hence $c = x^T A^{-1} G$. We also note that since A^{-1} is invertible, $\text{rank}(A^{-1}G) = \text{rank}(G) = k$, hence the k rows of $A^{-1}G$ are $\mathbb{F}_q$-linearly independent. ■

Definition 2.24. Let $\mathcal{C}$ be a $[n, k]_q$ code with generator matrix $G \in \mathbb{F}_q^{k\times n}$ and $I \subseteq \{1, \dots, n\}$ such that $|I| = k$. If G_I is invertible, then the I-indexed entries of $mG_I^{-1}G$ is called *information symbols* and I an *information set*.

Example 2.25. Consider the $[7, 4, 3]_2$ hamming code $\mathcal{H}_3$. That is the code with parity check matrix:

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

By Theorem 2.19.

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

is a parity check matrix for $\mathcal{H}_3$. Then obviously $I = \{1, 2, 3, 4\}$ is an information set since G_I is invertible. A less obvious information set would be $J = \{1, 3, 6, 7\}$ since:

$$G_J = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \sim I_{4\times 4}$$ □

Remark 2.26. If G is a generator matrix of $\mathcal{C}$, then so is $G^{-1}G$ and vice versa. This follows as the map $\mathbb{F}_q^k \ni x \mapsto x^T G_I^{-1} \in \mathbb{F}_q^n$ is bijective as $G_I^{-1} \in GL_n(\mathbb{F}_q)$. In addition $(G_I^{-1}G)_I = I_{k\times k}$.

The information symbols contains the information captured by m, and the $n - k$ symbols are the redundancy used for error correcting.

The basic idea behind information-set decoding (ISD), given a recived vector y is to find an information-set I such that the entries y_i with $i \in I$ are error free. If y_I is error free, we may obtain the we may obtain the original message as $y_I G_I^{-1}$, as $\text{rank}(G_I) = k$. Hence we will need algorithms for computing information sets given a generator matrix $G \in \mathbb{F}_q^{k\times n}$

2.4.1 Algorithms for Computing Information Sets

We start by introducing perhaps the simplest way to generate information sets in Algorithm 2.28. The idea behind the algorithm is stems from the following lemma:

Lemma 2.27. *Let $G \in \mathbb{F}_q^{k\times n}$ be the generator matrix of code $\mathcal{C}$, and G' be G reduced to row echelon form. Let I' be the tuple consisting of the indicies of the k pivot columns of G sorted with respect to the canonical order $\leq$ on $\mathbb{N}$, then any set of the form $\{i_1, i_2 \ldots, i_k\}$ with $i_j \in \mathbb{N}$ and $G'_{i_j,j} \neq 0$ such that $I'_j \leq i_j < I'_{j+1}$ for all $j < k$ and $I'_k \leq i_k \leq n$.*

Proof. Suppose $\{i_1, i_2 \ldots, i_k\}$ is such a set, then there exists a permutation matrix such that $P \in \mathbb{F}_q^{n\times n}$ such that the pivot columns in GP has indicies $i_1, i_2 \ldots, i_k$, since $G'_{i_j,j} \neq 0$. Hence the columns $g_{i_1}, g_{i_2} \ldots, g_{i_k}$ are $\mathbb{F}_q$-linearly independent and thus $\{i_1, i_2 \ldots, i_n\}$ also form an information-set. ■

The notion of a information set will become pivotal in the algorithms to come, hence we will need a efficient way to compute these information-sets, the procedures in Algorithm 2.28 provides a way to do exactly this, the correctness follows immediately by Lemma 2.27.

Algorithm 2.28 Construction of information sets using initial row reduction

procedure RR IS GENERATOR(G: a generator matrix of $[n, k]_q$ code, r: boolean value)
  $G' \leftarrow G$ reduced to row echelon form
  $I' \leftarrow$ The tuple consisting of the indicies of the k pivot columns in G'
  $J \leftarrow \left\{\{i_1, i_2 \ldots, i_k\} \subseteq \{1, 2, \ldots, n\} \,|G'_{i_j,j} \neq 0, I'_j \leq i_j < I_{j+1} \forall j < k, I'_k \leq i_k \leq n\right\}$
  if r **then**
    loop
      yield $I \in J$ ▷ Chosen uniformly, with replacement
  else
    for $I \in J$ **do**
      yield I ▷ Chosen uniformly, without replacement

Remark 2.29. Gaussian reduction, which is used compute G' has a time complexity of $O(nk \min\{n, k\}) = O(nk^2)$. In comparison the computational cost the other parts (looking up the pivot columns and constructing new information-sets given this information-set) of Algorithm 2.28, are negitable. Hence the bulk of the computation time will be spent when computing G'. This means that we may allow our algorithms to use multiple different information sets, with very little additional overhead.

Next we highlight a general weakness of Algorithm 2.28, namely that we aren't guaranteed to get every information set.

Example 2.30. Consider the generator matrix G of the $[7, 4, 3]_2$ hamming code $\mathcal{H}_3$ as described in Example 2.25. Then using the same notation as in Algorithm 2.28 we see that $G' = G$. Hence $I' = (1, 2, 3, 4)$. Next we find that

$$J = \{\{1, 2, 3, 4\}, \{1, 2, 3, 6\}, \{1, 2, 3, 7\}\}$$

Since $G_{5,4} = 0$. However J does not contain all of the information sets of $\mathcal{H}_3$, In particular J does not contain $\{1, 3, 6, 7\}$ which is also an information set, confer Example 2.25. □

Perhaps one way to combat this disadvantage could be to permute the columns of G via some permutation $\sigma \in S_n$, by letting:

$$P_\sigma = \left[p_{ij}\right] \text{ with } p_{ij} = \begin{cases} 1 & \text{if } \sigma(i) = j \\ 0 & \text{otherwise} \end{cases}$$

and apply the algorithm to GP_σ, obtaining the set J_σ containing some of the information sets of GP_σ. Then given $I_\sigma \in J_\sigma$ we may compute an information set for G via $\sigma^{-1}(I_\sigma)$. But we digress, instead we will consider picking a subset I of $\{1, 2, \ldots, n\}$ at random and checking if it forms an information set.

Algorithm 2.31 Constructing Information Sets using Gaussian Elimination

```
procedure GAUSSIAN IS GENERATOR(G: a generator matrix of [n, k]_q code,
                                r: boolean value)
    for I ⊆ {1, 2, ..., n} do        ▷ Chosen uniformly, with replacement if r is true.
        if G_I is invertible then
            yield I
```

Remark 2.32. One of the simplest ways to check if G_I is invertible in Algorithm 2.31, is to see if it is row equivalent with a upper triangular matrix (with non-zero entries in the diagonal). This can be done using Gaussian elimination.

The next algorithm is an improvement of Algorithm 2.31, the main idea is to gradually construct I by adding a new index at each iteration, which allows us to perform an early abort if one of the columns of G_I is linearly dependent on the rest.

Algorithm 2.33 Construction of information sets with early abort

```
procedure IS GENERATOR(G: a generator matrix of [n, k]_q code)
    loop
        I ← {i} with i ∈ {1, 2, ..., n}                          ▷ Picked uniformly
        while |I| ≠ k do
            J ← {j ∈ {1, 2, ..., n} | rank([G_I  G_{*,j}]) = |I| + 1}
            if |J| ≠ 0 then
                j ∈ J                                            ▷ Picked uniformly
                I ← I ∪ {j}
            else
                break
        if |I| = k then
            yield I
```

Each iteration of the while loops adds (if possible) a $j \in \{1, 2, \ldots, n\}$ to I such that the columns of G_I and $G_{*,j}$ are linearly independent, hence if the while-loop terminates with $|I| = k$, then we have found an information set. The other case is that the while-loop terminates but $|I| \neq k$, in that case we performed an early abort, since we couldn't find a $j \in \{1, 2, \ldots, n\}$ such that $\operatorname{rank}\left(\begin{bmatrix} G_I & G_{*,j} \end{bmatrix}\right) = |I| + 1$.

2.4.2 Information Set Decoding

Finally we can state our first information-set decoding algorithm.

Algorithm 2.34 Plain information-set decoding

procedure PLAIN ISD(G: a generator matrix of $[n,k]_q$ code $\mathcal{C}$, y: recived word, t: number of errors)
  for $I \in$ IS GENERATOR$(G, false)$ **do**
    $m' \leftarrow y_I G_I^{-1}$
    if $\mathrm{wt}(y - m'G) \leq t$ **then**
      return m'

Proof of the correctness of Algorithm 2.34. Suppose $y = m^T G + e^T$, where $m \in \mathbb{F}_q^k$ and $e \in \mathbb{F}_q^n$ such that $\mathrm{wt}(e) = t$ with $t \leq \lfloor \frac{d-1}{2} \rfloor$, where d is the minimum distance of the code generated by G. Assuming that the coordinates of y_I are error free, then $m' := y_I G_I^{-1}$ equals m and hence $\mathrm{wt}(y - m'G) = t$, and we return m. Otherwise if y_I isn't error free, then $m' \neq m$, and hence $\mathrm{wt}(mG - m'G) \geq 2t$ as $\mathcal{C}$ has a minimum distance of at least $2t$, combining this with the fact that y differs from mG in exactly t positions we see if $\mathrm{wt}(y - m'G) \leq t$, then we know that m' equals m. ■

Next we will briefly discuss the expected work factor of Algorithm 2.34. We will assume that the indicies non zero indicies of the error vector are distributed uniformly. The probability of choosing an information-set, which is error free, is $\binom{n-t}{k}\binom{n}{k}^{-1}$, after all we have $n-t$ error free positions and $\binom{n}{k}$ choices of k indicies. In addition the basic matrix inversion algorithm has a time total work factor of about αk^3 for some small α. Hence the total expected work factor is about:

$$W = \alpha k^3 \binom{n}{k}\binom{n-t}{k}^{-1}$$

The $\binom{n}{k}\binom{n-t}{k}^{-1}$ comes from the fact, that we are computing the expected value of a stochastic variable which follows a geometric distribution with success probability of $\binom{n-t}{k}\binom{n}{k}^{-1}$.

Lee-Brickell's Algorithm

The next algorithm, due to Lee and Brickell is a natural generalization of plain information set decoding, which allows for a number of errors p in the recived word y, more specifically given an information set I we allow y_I to have p corrupted entries.

Algorithm 2.35 Lee-Brickell's algorithm for information set decoding

procedure LEE-BRICKELL ISD(G: a generator matrix of $[n,k]_q$ code, y: recived word, t: number of errors, p: an integer such that $0 \leq p \leq t$)
  for $I \in$ IS GENERATOR$(G, false)$ **do**
    $y' \leftarrow y - y_I G_I^{-1} G$
    for $J = \{j_1, j_2 \ldots, j_p\} \subseteq I$ such that $|J| = p$ **do**
      for $m \in (\mathbb{F}_q^*)^p$ **do**
        $e \leftarrow y' - \sum_{i=1}^{p} m_i g_{*,j_i}$
        if $\mathrm{wt}(e) = t$ **then**
          return e

Since we have $\binom{k}{p}$ subsets J of I such that $|J| = p$, we should keep the parameter p relatively small compared to k. what about the binary case

When we pick $J \subseteq I$ such that $|J| = p$ and check if $\mathrm{wt}(y' - \sum_{i=1}^{p} m_i g_{j_i}) = t$ for all $m \in (\mathbb{F}_q^*)^p$, we are checking if there is exactly p symbols in y_I which are corrupted. This gives rise to a natural question, mainly: "Why not simply check the condition for all $m \in \mathbb{F}_q^p$, instead of $m \in (\mathbb{F}_q^*)^p$?" As this would allow for decoding y as long as y_I has p or fewer errors. It is a matter of minimizing the expected work, if we chose $m \in \mathbb{F}_q^p$ the inner loop in Algorithm 2.35 would have to perform q^p iterations. On the otherhand picking $m \in (\mathbb{F}_q^*)^p$ the loop only have to perform $(q-1)^p$ iterations. Additionally as we noted in Remark 2.29, the overhead of the process of finding an information set, after we have computed the first information set, is very low.

For a given information set I the probability for the algorithm successfully decoding y is:

$$\mathbb{P}(|I \cap \mathrm{supp}(e)| = p) = \binom{n-k}{t-p}\binom{k}{p}\binom{n}{t}^{-1}$$

Since we must have $t - p$ errors in the symbols of $y_{\{1,\dots,n\}\setminus I}$ and $\binom{n}{t}$ possible error vectors. Additionally the work performed iteration, given that it is unsuccessful is about:

$$W_I = \alpha k^3 \binom{k}{p}(q-1)^p$$

Since we have to compute G_I^{-1}, and $G_I^{-1}G$ have to chose $J \subseteq I$ such that $|J| = p$ and for each J we compute $y' - \sum_{i=1}^{p} m_i g_{*,j_i}$, with $m \in (\mathbb{F}_q^*)^p$. Combining these facts we get that the expected work factor of Lee-Brickell's algorithm is about:

$$W = \alpha k^3 \binom{k}{p}(q-1)^p \frac{\binom{n}{t}}{\binom{n-k}{t-p}\binom{k}{p}} = \alpha k^3 (q-1)^p \frac{\binom{n}{t}}{\binom{n-k}{t-p}}$$

again, since we are computing the expected value of a geometric distribution.

Sterns Algorithm

During this subsection we fix a $[n,k]_q$ code $\mathcal{C}$ furthermore for the sake of simplicity we assume that k is even. Let G be a generator matrix of $\mathcal{C}$. The main idea of Stern's algorithm is to split G into two sub matrices each with p rows, and check if the linear combinations of said rows, overlap at certain indices. If we find a match, then we the weight of the remaining (non-overlapping) part. If it has weight t then we have found error vector.

Algorithm 2.36 Stern's algorithm for information set decoding

procedure STERN ISD(G: a generator matrix of $\mathcal{C}$, ℓ: an integer with $0 \leq \ell \leq n-k$, p an integer with $0 \leq p \leq t$, y: received word with a maximum of t errors)

loop

$I \in$ IS GENERATOR$(G, true)$ ▷ Chosen with replacement

$y' \leftarrow y - y_I G_I^{-1} G$

Choose $X \subseteq I$ uniformly such that $|X| = k/2$

$Y \leftarrow I \setminus X$

Choose $Z \subseteq \{1, 2, \ldots, n\} \setminus I$ such that $|Z| = \ell$

for $A = \{a_1, a_2, \ldots, a_p\} \subseteq X$, $B = \{b_1, b_2, \ldots, b_p\} \subseteq Y$ with $|A| = |B| = p$ **do**

$(\mathcal{V}_A, \mathcal{V}_B) \leftarrow \left(\left\{y - \sum_{i=1}^{p} m_i g_{*,a_i} | m \in (\mathbb{F}_q^*)^p\right\}, \left\{y - \sum_{i=1}^{p} m_i g_{*,b_i} | m \in (\mathbb{F}_q^*)^p\right\}\right)$

for $a \in \mathcal{V}_A, b \in \mathcal{V}_B$ with $a_i = b_i$ for all $i \in Z$ **do**

$e \leftarrow a - b$

if $\mathrm{wt}(e) = t$ **then**

return $y - e$

Remark 2.37. Contrary to the approach in Algorithms 2.34 and 2.35. The information sets I, in Algorithm 2.36, are chosen with replacement, since $X, Y \subseteq I$ are choosen uniformly, hence we cannot simply iterate through each information set once. (Since we may not pick the "correct" sets X and Y for a given iteration)

We will not discuss the expected work factor of Sterns algorithm, instead we refer to Peters (2010)[Sections 4 and 5].

3 Algebraic Geometry Codes

This chapter will deal with the decoding of algebraic geometry codes, and subsequent construction of McElice PKC on these codes. Recall that an algebraic geometry code is defined as follows:

Definition 3.1. Let $\mathcal{X}$ be a regular absolutely irreducible protective algebraic curve of genus g. Furthermore let $\mathcal{P} = (P_1, P_2, \ldots, P_n)$ such that $P_1, P_2, \ldots, P_n \in \mathcal{X}$ are n distinct rational points. Let $D = \sum_{i=1}^{n} P_i$ and G be a rational divisor on $\mathcal{X}$ such that $\operatorname{supp}(D) \cap \operatorname{supp}(G) = \emptyset$, then the code $\mathcal{C}_L(\mathcal{X}, D, G) = \operatorname{Ev}_{\mathcal{P}}(L(G))$ is called an *algebraic geometry code* or simply an AG code.

Next we list a couple of results on the properties of these algebraic geometry codes, for the proofs of the properties we refer the reader to Nørbjerg (2023)[Chapter 3].

Theorem 3.2. *Let $\mathcal{C}_L(\mathcal{X}, G, D)$ be an algebraic geometry code and let $\mathcal{X}$ have genus g. Then the minimum distance d and dimension k of $\mathcal{C}_L(\mathcal{X}, G, D)$ satisfies:*

(i) $k = \ell(G) - \ell(G - D)$

(ii) $d \geq n - \deg(G)$. *For this reason $d^* = n - \deg(G)$ is called the designed minimum distance of $\mathcal{C}_L(\mathcal{X}, D, G)$*

(iii) *If $\deg(G) < n$, then $k = \ell(G)$. Furthermore if $f_1, f_2, \ldots, f_k \in L(G)$ is a $\mathbb{F}_q$-basis of $L(G)$, then*

$$M := \begin{bmatrix} f_1(P_1) & f_1(P_2) & \cdots & f_1(P_n) \\ f_2(P_1) & f_2(P_2) & \cdots & f_2(P_n) \\ \vdots & \vdots & \ddots & \vdots \\ f_k(P_1) & f_k(P_2) & \cdots & f_k(P_n) \end{bmatrix}$$

is a generator matrix of $\mathcal{C}_L(\mathcal{X}, D, G)$.

(iv) *If $2g - 2 < \deg(G) < n$, then $k = \deg(G) - g + 1$.*

Two natural practical problems comes to mind:

(i) Given a divisor G on $\mathcal{X}$ we need a way to compute a $\mathbb{F}_q$-basis of $L(G)$. This will give us the ability to compute a generator matrix for our algebraic geometry codes and will also be needed for most decoding algorithms.

(ii) Given an algebraic geometry code $\mathcal{C}_L(\mathcal{X}, D, G)$ we need an efficent decoding algorithm.

We will not address the first problem during the project. Instead we simply note that such an algorithm exists, we will however address the second problem in Chapter 4. However we will first introduce some more theory on algebraic geometry codes.

3.1 Differentials on Projective Algebraic Curves

The following section will be based on Pellikaan et al. (2018)[Sections 11.1.6 and 11.2.2] and Couvreur and Randriambololona (2020)[Chapter 3].

Definition 3.3. Let $\mathcal{X}$ be a plane irreducible regular projective curve over $\mathbb{F}$ and $\mathcal{V}$ be a vectorspace over $\mathbb{F}(\mathcal{X})$. A $\mathbb{F}$-linear map $D : \mathbb{F}(\mathcal{X}) \to \mathcal{V}$ is called a *derivation* if it satisfies the *Leibniz rule*:

$$D(fg) = fD(g) + gD(f)$$

for all $f, g \in \mathcal{V}$.

Remark 3.4. If $D : \mathbb{F}(\mathcal{X}) \to \mathcal{V}$ is a derivation, then:

$$D(1) = D(1 \cdot 1) = D(1) + D(1)$$

meaning $D(1) = 0$.

Example 3.5. The projecitve line $\mathbb{P}^1$, with defining equation $Y = 0$, has the function field $\mathbb{F}(X)$. Let $F = \sum_{i=0}^{m} a_i X^i$ be a polynomial in $\mathbb{F}[X]$, then define $D(F) = \sum_{i=1}^{m} i a_i X^{i-1}$. Extending D to $\mathbb{F}(\mathcal{X})$ by defining:

$$D\left(\frac{F}{G}\right) = \frac{GD(F) - FD(G)}{G^2}$$

yields an gives an derivation $D : \mathbb{F}(\mathcal{X}) \to \mathbb{F}(\mathcal{X})$ since:

$$\begin{aligned}
D\left(\frac{F_1}{G_1}\frac{F_2}{G_2}\right) &= \frac{G_1G_2D(F_1F_2) - F_1F_2D(G_1G_2)}{(G_1G_2)^2} \\
&= \frac{G_1G_2(F_1D(F_2) + F_2D(F_1)) - F_1F_2(G_1D(G_2) + G_2D(G_1))}{(G_1G_2)^2} \\
&= \frac{G_2F_1D(F_2) - F_1F_2D(G_2)}{G_1G_2^2} + \frac{G_1F_2D(F_1) - F_1F_2D(G_1)}{G_1^2G_2} \\
&= \frac{F_1}{G_1}\underbrace{\frac{G_2D(F_2) - F_2D(G_2)}{G_2^2}}_{=D(F_2/G_2)} + \frac{F_2}{G_2}\underbrace{\frac{G_1D(F_1) - F_1D(G_1)}{G_1^2}}_{=D(F_1/G_1)}
\end{aligned}$$

holds for all $\frac{F_1}{G_1}, \frac{F_2}{G_2} \in \mathbb{F}(\mathcal{X})$. □

Definition 3.6. Let $Der(\mathcal{X}, \mathcal{V})$ denote the set of all derivations $D : \mathbb{F}(\mathcal{X}) \to \mathcal{V}$, additionally we denote $Der(\mathcal{X}, \mathbb{F}(\mathcal{X}))$ by $Der(\mathcal{X})$. A $\mathbb{F}(\mathcal{X})$-linear transformation from $Der(\mathcal{X})$ to $\mathbb{F}(\mathcal{X})$ is called a *differential* on $\mathcal{X}$, the set of differnetials on $\mathcal{X}$ is denoted by $\Omega(\mathcal{X})$.

Let $\mathcal{V} \subseteq \mathbb{F}(\mathcal{X})$ be a $\mathbb{F}(\mathcal{X})$-subspace, then both $Der(\mathcal{X}, \mathcal{V})$ and $\Omega(\mathcal{X})$ forms $\mathbb{F}(\mathcal{X})$-vectorspaces.

The fact that $Der(\mathcal{X}, \mathcal{V})$ forms a $\mathbb{F}(\mathcal{X})$-vectorspace can be seen as follows: Let $D_1, D_2 \in Der(\mathcal{X}, \mathcal{V})$, then we define their sum as by $(D_1 + D_2)(f) = D_1(f) + D_2(f)$, now since both D_1 and D_2 satisfies the Leibniz rule, then so will $(D_1 + D_2)$. Additionally for $f \in \mathbb{F}(\mathcal{X})$ and $D \in Der(\mathcal{X}, \mathcal{V})$ we define their multilication as $(fD)(g) = fD(g)$.

Lemma 3.7. *Let $D \in Der(\mathcal{X}, \mathcal{V})$ then if there exists an $f \in \mathbb{F}(\mathcal{X})$ such that $D(f) = 1$, then $D(f^k) = \phi(k)f^{k-1}$ for all $k \in \mathbb{N} \setminus \{0\}$, where ϕ is the unique ring homomorphism from $\mathbb{Z}$ to $\mathbb{F}(\mathcal{X})$.*

Proof. We shall prove the result using induction. Thus suppose $k = 1$ we see that

$$D(f) = 1 = \phi(1)f^0$$

Next consider $k \in \mathbb{N}$ such that $k > 1$, then:

$$\begin{aligned} D(f^k) = D(f)f^{k-1} + D(f^{k-1})f &\overset{(a)}{=} f^{k-1} + (\phi(k-1)f^{k-2})f \\ &= (\phi(1) + \phi(k-1))f^{k-1} = \phi(k)f^{k-1} \end{aligned}$$

where equality (a) follows from the induction hypothesis. ∎

Theorem 3.8. *Let t be a local parameter at $P \in \mathcal{X}$. Then there exists a unique derivation $D_t : \mathbb{F}(\mathcal{X}) \to \mathbb{F}(\mathcal{X})$ such that $D_t(t) = 1$.*

Proof. Let $f, g \in \mathbb{F}(\mathcal{X})$, since t is a local parameter at P there exists $u, v \in \mathcal{O}_P(\mathcal{X})$ such that $f = ut^m$ and $g = vt^n$ where $m = v_P(f)$ and $n = v_P(g)$ respectively. Suppose that $D_t : \mathbb{F}(\mathcal{X}) \to \mathbb{F}(\mathcal{X})$ such that $D_t(t) = 1$, then:

$$D_t(f) = D_t(ut^m) = D_t(u)t^m + uD_t(t^m) \overset{(a)}{=} u\phi(m)t^{m-1}$$

where ϕ denotes the unique ring homomorphism between $\mathbb{Z}$ and $\mathbb{F}(\mathcal{X})$. Equality (a) follows by Lemma 3.7 and the fact that $D(u) = 0$ since $u \in \mathcal{O}_P(\mathcal{X})$ could you perhaps elaborate?.

Next for the existence of such a derivation: We let $D_t : \mathbb{F}(\mathcal{X}) \to \mathbb{F}(\mathcal{X})$ be the derivation defined as: $D_t(f) = D_t(ut^m) := u\phi(m)t^{m-1}$. It is easy to verify that D_t is $\mathbb{F}$-linear, since $D_t((cu)t^m) = (cu)\phi(m)t^{m-1} = cD_t(ut^m)$. Next consider $D_t(f+g)$ without loss of generality we may assume that $m = v_P(f) \leq v_P(G) = n$, thus:

$$D_t(f+g) = D_t((ut^{m-n} + v)t^n) = (ut^{m-n} + v)\phi(n)t^{n-1} = \underbrace{u\phi(n)t^{m-1}}_{\neq D_t(f)} + \underbrace{v\phi(n)t^{m-1}}_{=D_t(g)}$$

since we generally do not have $\phi(n) = \phi(m)$ The above statement is a problem since a derivation is supposed to be $\mathbb{F}$ linear ∎

The next corollary is a natural consequence of Theorem 3.8, which allows us to find a $\mathbb{F}(\mathcal{X})$-basis of $Der(\mathcal{X})$.

Corollary 3.9. *Let t be a local parameter at $P \in \mathcal{X}$ and D_t be the unique derivation such that $D_t(t) = 1$, then D_t forms a $\mathbb{F}(\mathcal{X})$-basis for $Der(\mathcal{X})$ for all local parameters t.*

Proof. Suppose $D \in Der(\mathcal{X})$, with $D(t) = f$ for some $f \in \mathbb{F}(\mathcal{X})$. Then $f^{-1}D \in Der(\mathcal{X})$, since $Der(\mathcal{X})$ forms a $\mathbb{F}(\mathcal{X})$-vectorspace. Additionally as $(f^{-1}D)(t) = 1$, we see that $D_t = f^{-1}D$, by Theorem 3.8, and hence $fD_t = D$. ∎

Let $f \in \mathbb{F}(\mathcal{X})$, then we define the map $df : Der(\mathcal{X}) \to \mathbb{F}(\mathcal{X})$ by $df(D) = D(f)$.

Theorem 3.10. *If $t \in \mathbb{F}(\mathcal{X})$ is a local parameter at $P \in \mathcal{X}$, then dt is a $\mathbb{F}(\mathcal{X})$-basis of $\Omega(\mathcal{X})$.*

Proof. Let $\omega \in \Omega(\mathcal{X})$ and $D \in Der(\mathcal{X})$ then $D = D(t)D_t$ by the proof of Corollary 3.9. We see that:

$$\omega(D) = \omega(D(t)D_t) = D(t)\omega(D_t) = dt(D)\omega(D_t)$$

Hence $\omega = fdt$ where $f = d(D_t) \in \mathbb{F}(\mathcal{X})$. ■

Definition 3.11. Let $\omega \in \Omega(\mathcal{X})$ then $\omega = fdt$ for some local parameter t of $P \in \mathcal{X}$, then we define $\text{ord}_P(\omega) = \text{ord}_P(f)$. If $\omega \neq 0$ then we define the *canonical divisor* associated with ω as:

$$(\omega) := \sum_{P \in \mathcal{X}} ord_P(\omega)$$

Definition 3.11, raises some natural questions, namely:

(i) Is $\text{ord}_P(\omega)$, well defined (meaning does it is independent on our choice of f). Hence we will assume that t_1 and t_2 is local parameters at P, then there exists $f_1, f_2 \in \mathbb{F}(\mathcal{X})$ such that $\omega = f_1 dt_1 = f_2 dt_2$.

(ii) Is the divior (ω) well defined Passer denne forklaring eller arbejder vi over $\overline{\mathbb{F}_q}$(meaning is $\text{ord}_P(\omega) = 0$ for all but a finite number of points $P \in \mathcal{X}$. We will not discuss this over a general field $\mathbb{F}$, instead we refer the reader to Fulton (2008)[Section 8.5]. We will however consider the case where $\mathbb{F} = \mathbb{F}_q$, hence $|\mathcal{X}| \leq |\mathbb{P}^n(\mathbb{F}_q)| \leq n \left|\mathbb{F}_q^{n-1}\right|$, when $\mathcal{X} \subseteq \mathbb{P}^n(\mathbb{F}_q)$. Where the last inequality follows since there is $(q^{n-1}) = \left|\mathbb{F}_q^{n-1}\right|$ distinct points of the form $P = (1 : P_1 : \cdots : P_n) \in \mathbb{P}^n(\mathbb{F}_q)$.)

Proposition 3.12. *Let $\mathcal{X}$ be a projective regular curve of genus g over an algebraically closed field $\mathbb{F}$ and $\omega \in \Omega(\mathcal{X})$ then* $\deg((\omega)) = 2g - 2$.

One may think that this follows more or less directly from the Reimann Roch Theorem, however this approach forms a circular argument as the statement of Proposition 3.12 is used to prove the Reimann Roch Theorem, see for instance Fulton (2008)[Section 8.5][1]. Hence we will refer to Fulton (2008)[Section 8.3] for a proof of the result, since the theory of canonical divisors and differentials is not the main objects of study in this project.

Definition 3.13. Let D be a divisor on $\mathcal{X}$, then we define:

$$\Omega(D) = \{\omega \in \Omega(\mathcal{X}) | (\omega) - D \text{ is effective}\}$$

The space $\Omega(D)$ forms a $\mathbb{F}$-vectorspace, just like the Riemann-Roch space $L(D)$, the dimension of $\Omega(D)$ will be denoted by $\delta(D)$.

3.1.1 Residue Codes

Suppose $f/g \in \mathbb{F}(\mathcal{X})$ and that t is a local parmeter at $P \in \mathcal{X}$. Recall that $v_P(f/g) = v_P(f) - v_P(g)$. Let $z \in \mathbb{F}(\mathcal{X})$ if $m := v_P(z)$, then we can write $z = at^m + z'$ where $a \in \mathbb{F}^*$ and $v_P(z') > m$ similarly z' can be expanded in a similar way, hence we may write:

$$z = \sum_{i \geq m} a_i t^i$$

[1]There might be other approaches for proving the Reimann-Roch Theorem such as the approach in Stichtenoth (2009), in which case the discussed approach for proving Proposition 3.12 should work.

with $a_i \in \mathbb{F}$ and $a_m \neq 0$. Additionally we will use the convention that $a_i = 0$ if $i < v_P(f)$ to write:

$$z = \sum_i a_i t^i$$

This leads us to the following definition.

Definition 3.14. Let t be a local parameter at $P \in \mathcal{X}$ and $\omega = fdt$. The rational function f can be written as $\sum_i a_i t^i$ with $a_i \in \mathbb{F}$. We call a_{-1} the *residue* of ω at P and denote it by $\mathrm{Res}_P(\omega)$.

The residue $\mathrm{Res}_P(\omega)$ doesn't depend on the chosen local parameter,

Definition 3.15. Let $\mathcal{X}$ be a projective absolutely irreducible regular curve over $\mathbb{F}_q$ and let $\mathcal{P} := (P_1, P_2, \ldots, P_n)$ be a tuple of n distinct rational points of $\mathcal{X}$. Define $D := \sum_{i=1}^n P_i$ and let G be a divisor on $\mathcal{X}$ such that $\mathrm{supp}(G) \cap \mathrm{supp}(D) = \emptyset$. Furthermore define $\mathrm{Res}_{\mathcal{P}} : \Omega(G - D) \to \mathbb{F}_q^n$ as:

$$\mathrm{Res}_{\mathcal{P}}(\omega) = (\mathrm{Res}_{P_1}(\omega), \mathrm{Res}_{P_2}(\omega), \ldots, \mathrm{Res}_{P_n(\omega)})$$

then we define the *residue code* $\mathcal{C}_\Omega(\mathcal{X}, D, G)$ as $\mathcal{C}_\Omega(\mathcal{X}, D, G) := \mathrm{Res}_{\mathcal{P}}(\Omega(G - D))$, the constant $d^* = \deg(G) + 2 - 2g$, where g is the genus of $\mathcal{X}$, is called the *designed minimum distance* of $\mathcal{C}_\Omega(\mathcal{X}, D, G)$..

The relationship between residue codes and evaluation codes, is fascinating[2], however we will only mention the following theorem.

Theorem 3.16. *Let $P_1, P_2, \ldots, P_n$ be distinct points on the projective plane curve $\mathbb{P}_1$, such that none of the points lie at infinity, meaning $P_i = (x_i : 1)$ for some $x_i \in \mathbb{F}_q^*$ for all $i = 1, 2, \ldots, n$. Furthermore let $D = \sum_{i=1}^n P_i$ and G be a divisor on $\mathbb{P}^1$ such that $\mathrm{supp}(D) \cap \mathrm{supp}(G) = \emptyset$. Then:*

$$\mathcal{C}_\Omega(\mathbb{P}^1, D, G) = \mathcal{C}_L(\mathbb{P}^1, D, (\omega) + D - G)$$

with $\omega := \frac{dh}{h}$ where $h(x) := \prod_{i=1}^n (x - x_i)$.

The proof of the theorem can be found in Couvreur and Randriambololona (2020)[Proposition 35][3]

Corollary 3.17. *Under the assumptions of Theorem 3.16 the minimum distance of $\mathcal{C}_\Omega(\mathbb{P}^1, D, G)$ satisfies:*

$$d(\mathcal{C}_\Omega(\mathbb{P}^1, D, G)) \geq d^* = \deg(G) + 2 \tag{3.1}$$

Furthermore if $2g - 2 < \deg(G) < n$, then:

$$\dim_{\mathbb{F}_q}(\mathcal{C}_\Omega(\mathbb{P}^1, D, G)) = n - \deg(G) - 1 \tag{3.2}$$

[2]For example the dual code of an evaluation code is a residue code, and vice versa.

[3]Be aware that they state that $\mathcal{C}_\Omega(\mathbb{P}^1, D, G) = \mathcal{C}_L(\mathbb{P}^1, D, (\omega) - D + G)$, however they do inded prove the theorem which is stateted above.

Proof. From Theorem 3.16 we have $\mathcal{C}_\Omega(\mathbb{P}^1, D, G) = \mathcal{C}_L(\mathbb{P}^1, D, (\omega) + D - G)$. The inequality in Equation (3.1) now follows as:

$$\begin{aligned} d(\mathcal{C}_L(\mathcal{X}, D, (\omega) + D - G)) &= n - \deg((\omega) + D - G) \\ &= n - \deg((\omega)) - \deg(D) + \deg(G) \\ &= n - (2g - 2) + n + \deg(G) \\ &= \deg(G) + 2 \end{aligned}$$

by Proposition 3.12 additionally we note that deg : $Div(\mathbb{P}^1) \to \mathbb{N}$ is a homomorphism and that the genus of $\mathbb{P}^1$ is 0. Next Equation (3.2) follows under the assumption that $2g - 2 < \deg(G) < n$ as this implies $2g - 2 < \deg((\omega) + D - G) = 2g - 2 + n - \deg(G) < n$ which in turn implies that:

$$\begin{aligned} \dim_{\mathbb{F}_q}(\mathcal{C}_L(\mathbb{P}^1, D, (\omega) + D - G)) &\overset{(b)}{=} \deg((\omega) + D - G) - g + 1 \\ &\overset{(b)}{=} -2 + n - \deg(G) + 1 \\ &= n - \deg(G) - 1 \end{aligned}$$

where (a) follows by Theorem 3.2(iv) and (b) by Proposition 3.12 and the fact that the genus of $\mathbb{P}^1$ is 0. ■

Remark 3.18. We will restrict our selves to residue codes of the form $\mathcal{C}_\Omega(\mathbb{P}^1, D, G)$. Hence Theorem 3.16 implies that it is sufficient for us to describe decoding algorithms for AG codes of the form $\mathcal{C}_L(\mathcal{X}, D, G)$.

Theorem 3.19. *Let $\mathcal{X}$ be a smooth projective algebraic curve and ω be a differential on $\mathcal{X}$, then:*

$$\sum_{P \in \mathcal{X}} \mathrm{Res}_P(\omega) = 0$$

We will not provide a proof, instead we refer to Stichtenoth (2009)[Proposition 4.3.3].

3.2 Weirstrass Gaps

4 Decoding of AG Codes

Throughout this section we will fix a algebraic geometry code $\mathcal{C}_{D,G}$ constructed on the curve $\mathcal{X}$ of genus g. We will describe and show the correctness of an t-error decoding algorithm for $\mathcal{C}_{D,G}$ with

$$t \leq \left\lfloor \frac{d^* - 1}{2} - \frac{g}{2} \right\rfloor$$

The term $\frac{g}{2}$ is called the *genus penalty*. Our treatment will be based on Couvreur and Randriambololona (2020)[Section 6.1]

We start by proving the following proposition, which will guide our search for such an decoding algorithm.

Proposition 4.1. *Let $\mathcal{C}$ be a $[n, k, d]_q$ code with parity check matrix H. Furthermore let $y = c + e$ where $c \in \mathcal{C}$ and $e \in \mathbb{F}_q^n$ and $J \subseteq \{1, \ldots, n\}$, such that* $\operatorname{supp}(e) \subseteq J$ *and $|J| < d$. Then e is the unique solution to the system:*

$$\begin{cases} He = Hy \\ e_i = 0 \text{ for all } i \in \{1, \ldots, n\} \setminus J \end{cases} \tag{4.1}$$

Proof. Clearly e is a solution to the system in Equation (4.1) since $Hy = Hc + He = He$. Additionally if e' is another solution to $He' = Hy$ such that $\operatorname{supp}(e) \subseteq J$, then $e - e' \in \operatorname{null}(H) = \mathcal{C}$, however as $|J| < d$ the codeword $e' - e$ has weight $\operatorname{wt}(e' - e) \leq d$ meaning $e' - e = 0$. ∎

For the remaining part of the section we will let $y = c + e$ with $c \in \mathcal{C}_{D,G}$, meaning $c = (f(P_1), f(P_2), \ldots, f(P_n))$ for some $f \in L(G)$, and $e \in \mathbb{F}_q^n$ such that $\operatorname{wt}(e) \leq t$, unless otherwise specified. Proposition 4.1, implies that if we can find a small subset J such that $\operatorname{supp}(e) \subseteq J$, correcting the errors in the recived vector y is simply a matter of solving the linear system presented in Equation (4.1) for $e \in \mathbb{F}_q^n$. To find such a subset J we introduce the concept of an *error locating function.*

We will now describe a procedure for locating the errors. Let F be a divisor such that $\operatorname{supp}(F) \cap \operatorname{supp}(D) = \emptyset$. If $\lambda \in L(F)$ vanishes at every point P_i where $i \in \operatorname{supp}(e)$ then:

$$\lambda(P_i) y_i = \lambda(P_i) f(P_i) \text{ for all } i \in \{1, \ldots, n\}.$$

Since either $e_i = 0$ meaning $y_i = f(P_i)$ or alternatively if $i \in \operatorname{supp}(e)$ then $\lambda(P_i) = 0$. Hence λ can be used to locate the $\operatorname{supp}(e)$.

Remark 4.2. If $f \in L(G)$ and $\lambda \in L(F)$, then $\lambda f \in L(G + F)$. This can be seen as follows

$$(\lambda f) = \sum_{P \in \mathcal{X}} v_P(\lambda f) P \overset{(a)}{=} \sum_{P \in \mathcal{X}} (v_P(\lambda) + v_P(f)) P = (\lambda) + (f)$$

where (a) follows as v_P is a discrete valuation. Now since $(f)+G$ and $(\lambda)+F$ are effective divisors, we see that $(\lambda f)+(G+F)=(\lambda)+(f)+G+F$ is effective.

The contents of Remark 4.2, motivates the following definition:

Definition 4.3. Let F be a divisor on $\mathcal{X}$ such that $\operatorname{supp}(F)\cap\operatorname{supp}(D)=\emptyset$, then we define the set of *error locating functions* as:

$$K_y(F) := \{\lambda \in L(F) \mid (\lambda(P_1)y_1, \lambda(P_2)y_2, \ldots, \lambda(P_n)y_n) \in \mathcal{C}_{D,G+F}\}$$

By now we can finally state the basic algorithm.

Algorithm 4.4 Basic Decoding Algorithm

procedure BASIC DECODING(y: received word with a maximum of t errors
$f_1, f_2, \ldots, f_m$: a basis for $L(F)$,
(H, H'): parity check matrices for $\mathcal{C}_{D,G}$ and $\mathcal{C}_{D,G+F}$)
Compute $K_y(F)$
if $K_y(F)=\{0\}$ **then**
return ?
else
▷ Both $f_1, f_2, \ldots, f_m$ and H' are used implicitly to find $\lambda \in K_y(F)\setminus\{0\}$.
$J \leftarrow \{i \in \{1,\ldots,n\} | \lambda(P_i)=0\}$ for some $\lambda \in K_y(F)\setminus\{0\}$
$S \leftarrow \left\{e \in \mathbb{F}_q^n \middle| He = Hy \text{ and } e_i = 0 \text{ for all } i \in \{1,\ldots,n\}\setminus J\right\}$
if $|S| \neq 1$ **then**
return ?
else
return $y-e$ where e is the unique solution in S.

Next we will show that correctness of Algorithm 4.4, we start by proving the following lemma.

Lemma 4.5. *Let* $D_e = \sum_{i\in\operatorname{supp}(e)} P_i$. *If* $t \leq d^* - \deg(F) - 1$, *then* $K_y(F) = L(F - D_e)$.

Proof. We start by showing that $K_y(F) \supseteq L(F-D_e)$. So assume that $\lambda \in L(F-D_e)$. Then $\lambda(P_i)=0$ for all $i \in \operatorname{supp}(e)$, as $\operatorname{supp}(F)\cap\operatorname{supp}(D_e) \subseteq \operatorname{supp}(F)\cap\operatorname{supp}(D)=\emptyset$, hence $\lambda(P_i)y_i = \lambda(P_i)f(P_i)$ for all $i \in \{1,\ldots,n\}$, hence $(\lambda(P_1)y_1, \lambda(P_2)y_2, \ldots, \lambda(P_n)y_n) \in \mathcal{C}_{D,G+F}$ as $\lambda f \in L(G+F)$, by Remark 4.2.

Next we show that $K_y(F) \subseteq L(F-D_e)$. Let $c_y = (\lambda(P_1)y_1, \lambda(P_2)y_2, \ldots \lambda(P_n)y_n) \in \mathcal{C}_{D,G+F}$. Now since $\lambda f \in L(G+F)$, we see that $c_f = (\lambda(P_1)f(P_1), \lambda(P_2)f(P_2), \ldots, \lambda(P_n)f(P_n)) \in \mathcal{C}_{D,G+F}$. Since $\mathcal{C}_{D,G+F}$ is a linear subspace, we see that:

$$c_e = c_y - c_f = (\lambda(P_1)e_1, \lambda(P_2)e_2, \ldots, \lambda(P_n)e_n) \in \mathcal{C}_{D,G+F}$$

additionally we note that $\operatorname{wt}(c_e) \leq \operatorname{wt}(e) \leq t$. Now applying Theorem 3.2(ii) we see that the minimum distance of $\mathcal{C}_{D,G+F}$ is at least $n-\deg(G+F)$. However $\operatorname{wt}(c_e) \leq t \leq d^* - \deg(F) - 1 = n - \deg(G+F) - 1 < n - \deg(G+F)$. Hence $c_e = 0$ and $\lambda(P_i)=0$ for all $i \in \operatorname{supp}(e)$, so $\lambda \in L(F-D_e)$. ■

By now we are finally able to state and prove the correctness of the basic decoding algorithm.

Theorem 4.6. *Let $\mathcal{C}_L(\mathcal{X}, D, G)$ be a $[n, k, d]$ AG-code with designed minimum distance d^*. Furthermore let F be a divisor on $\mathcal{X}$ such that $\text{supp}(F) \cap \text{supp}(D) = \emptyset$ and $\deg(F) = t + g$ where $t \leq \left\lfloor \frac{d^*-1}{2} - \frac{g}{2} \right\rfloor$ and g is the genus of $\mathcal{X}$. Then Algorithm 4.4 is a t-error decoding algorithm for $\mathcal{C}_L(\mathcal{X}, D, G)$. Furthermore if $\deg(F) < n$ then the algorithm returns the good solution in $O(n^\omega)$ operations in $\mathbb{F}_q$ with $\omega \leq 3$.*

Proof. **TODO** der er noget iffy med det her bevis. I starten (altså før det med tidskompleksitet). The condition $t \leq \left\lfloor \frac{d^*-1}{2} - \frac{g}{2} \right\rfloor$ implies that

$$2t + g \leq d^* - 1$$

which in turn implies that $t \leq d^* - \deg(F) - 1$, applying Lemma 4.5, we see $K_y(F) = L(F + D_e)$ and in particular that $K_y(F) \neq \{0\}$ as $\ell(F - D_e) > 0$, by Propsition B.1, as $\deg(F) \geq t + g$ implies $\deg(F - D_e) \geq g$ since $\deg(D_e) \leq t$. Hence one can take a non-zero rational function $\lambda \in K_y(F)$. Now as $\lambda \in K_y = L(F - D_e)$, we know that $\deg((\lambda)_0) \leq \deg(F)$. In addition we noted earlier that $t \leq d^* - \deg(F) - 1$, meaning $\deg((\lambda)_0) \leq \deg(F) \leq d^* - 1 < d^*$. So $|\ker(\lambda)| < d^* < d(C_{D,G})$. Applying Proposition 4.1, we see that there will exist a unique solution to the system: $He = Hy$ and $e_i = 0$ for all $i \in \{1, \ldots, n\}$ such that $\lambda(P_i) \neq 0$.
Next we show that Algorithm 4.4 has time complexity $O(n^\omega)$. We start by noting that solving a linear system such as $He = Hy$ under the constraint that $e_i = 0$ for all $i \in \{1, 2, \ldots, n\} \setminus J$ can be done in $O(n^\omega)$, with $\omega \leq 3$ (Should probaly have a source). We will show that finding a non-zero $\lambda \in K_y(F)$, boils down to solving a linear system of similar dimensions. Suppose $\lambda \in L(F)$, we note that $c_\lambda := (\lambda(P_1), \lambda(P_2), \ldots, \lambda(P_n)) * y \in \mathcal{C}_{D,G+F}$ if and only if:

$$0 = H'(c_\lambda * y) = \sum_{j=1}^{n} h'_j (c_\lambda)_j y_j = \underbrace{\begin{bmatrix} h'_1 y_1 & h'_2 y_2 & \cdots & h'_n y_n \end{bmatrix}}_{:= H'_y} c_\lambda$$

Additionally since $\lambda \in L(F)$, there exists some $a_1, a_2, \ldots, a_m \in \mathbb{F}_q$, where $m = \ell(F)$, such that $\lambda = \sum_{i=1}^{m} a_i f_i$ thus:

$$c_\lambda = \begin{bmatrix} f_1(P_1) & f_2(P_1) & \cdots & f_m(P_1) \\ f_1(P_2) & f_2(P_2) & \cdots & f_m(P_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(P_n) & f_2(P_n) & \cdots & f_m(P_n) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix}$$

Hence we can find a non-zero $\lambda \in K_y$ by finding a non-zero solution to:

$$H'_y \begin{bmatrix} f_1(P_1) & f_2(P_1) & \cdots & f_m(P_1) \\ f_1(P_2) & f_2(P_2) & \cdots & f_m(P_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(P_n) & f_2(P_n) & \cdots & f_m(P_n) \end{bmatrix} a = 0 \tag{4.2}$$

and setting $\lambda = \sum_{i=1}^{m} a_i f_i$, since a non-zero solution to Equation (4.2) can be found in $O(\max\{k, m\}^\omega)$. Considering the AG-code $\mathcal{C}_L(\mathcal{X}, D, F)$ applying Theorem 3.2(iii) we see that $\dim_{\mathbb{F}_q}(\mathcal{C}(\mathcal{X}, D, F)) = \ell(F) = m$. Finally since the length of $\mathcal{C}(\mathcal{X}, D, F)$ is n we see that $m < n$ and that a non-zero λ can be found using $O(n^\omega)$ operations. Thus the total time complexity of the basic algorithm is $O(n^\omega)$. ∎

4.1 Error Correcting Pairs

In this section we will discuss an generalization of the basic algorithm, to the level of codes. The algorithm is the error correcting pairs algorithm (ECP algorithm). The primary principal of the algorithm is the same (locate the errors, and subsequently correct them using Proposition 4.1). Our treatment will be based on Panaccione (2021)[Subsections 1.6.2 and 1.8.1] as well as Couvreur and Randriambololona (2020)[Subsection 6.1.2].

We start by noting that $\mathbb{F}_q^n$ is the product of n fields, hence it forms a ring, together with element wise addition and multiplication, this motivates the following definition:

Definition 4.7. Let $a, b \in \mathbb{F}_q^n$ then we define the *hadaman* product of a and b as:

$$a * b := (a_1 b_1, a_2 b_2, \ldots, a_n b_n)$$

Additionally if $\mathcal{A}, \mathcal{B} \subseteq \mathbb{F}_q^n$ are linear codes, we define their *hadaman* product as

$$\mathcal{A} * \mathcal{B} := \operatorname{span}_{\mathbb{F}_q} \{a * b | a \in \mathcal{A}, b \in \mathcal{B}\}$$

Remark 4.8. If $a, b, c \in \mathbb{F}_q^n$, then $\langle a * b, c\rangle = \langle a, b * c\rangle = \langle a * c, b\rangle$ where $\langle \cdot, \cdot \rangle$ denotes the canonical inner product in $\mathbb{F}_q^n$.

Lemma 4.9. *Let $\mathcal{A}, \mathcal{B}, \mathcal{C} \subseteq \mathbb{F}_q^n$ be linear codes, then $\mathcal{A} * \mathcal{B} \subseteq \mathcal{C}^\perp$ if and only if $\mathcal{A} * \mathcal{C} \subseteq \mathcal{B}^\perp$.*

Proof. Assume $\mathcal{A} * \mathcal{B} \subseteq \mathcal{C}^\perp$. Hence if $a \in \mathcal{A}$, $b \in \mathcal{B}$ and $c \in \mathcal{C}$, then $\langle a * b, c\rangle = 0$, the rest follows by combining this with Remark 4.8 which states that $\langle a * b, c\rangle = \langle a * c, b\rangle$. A similar argument can be constructed to show the other implication. ■

We need to introduce some special notation before we can describe the error correcting pairs algorithm.

Definition 4.10. Let $J = \{j_1, j_2, \ldots, j_m\} \subseteq \{1, \ldots, n\}$ and $x \in \mathbb{F}_q^n$. Then we let x_J denote $(x_{j_1}, x_{j_2}, \ldots, x_{j_m})$, that is the projection of x on the coordinates in J and let $Z(x) := \{i \in \{1, \ldots, n\} | x_i = 0\}$.
In addition for $A \subseteq \mathbb{F}_q^n$ we define:

(i) $A_J = \{a_J | a \in A\} \subseteq \mathbb{F}_q^{|J|}$ (extracting)
(ii) $A(J) := \{a \in A | a_J = 0\}$ (shortening)
(iii) $Z(A) := \{i \in \{1, \ldots, n\} | a_i = 0 \text{ for all } a \in A\}$

Remark 4.11. Let $\mathcal{C}$ be a $[n, k]_q$ code with generator matrix $G \in \mathbb{F}_q^{n \times k}$, then if I is an information set, then $\mathcal{C}_I = \mathbb{F}_q^k$, since $|I| = k$ and G_I is non-singular.

We note that this notation differs from the normal notation used in coding theory. For instance the shortening of set $A \subseteq \mathbb{F}_q^n$ normally referees the set $\{a_I | a \in A(J)\}$ where $I = \{1, 2, \ldots, n\} \setminus J$. We use this alternative notation because we want to be able to apply the hadaman product.

Example 4.12. Consider the $[2,2]_3$ code $\mathcal{C}$. With generator matrix: $G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 1 \end{bmatrix}$, Then:

$$\mathcal{C} = \{(0,0),(0,2,1),(0,1,2),(1,2,1),(1,1,2),(2,1,2)\}$$

hence $\mathcal{C}(\{1\}) = \{(0,0,0),(0,2,1),(0,1,2)\}$ while $\mathcal{C}_{\{2,3\}} = \{(0,0),(2,1),(1,2)\}$. □

Definition 4.13. Let $\mathcal{A},\mathcal{B},\mathcal{C} \subseteq \mathbb{F}_q^n$ be linear codes. Then the pair $(\mathcal{A},\mathcal{B})$ is called a *t-error correcting pair* for $\mathcal{C}$ if the following conditions are satisfied:

(ECP1) $\mathcal{A} * \mathcal{B} \subseteq \mathcal{C}^\perp$
(ECP2) $\dim_{\mathbb{F}_q}(\mathcal{A}) > t$
(ECP3) $d(\mathcal{B}^\perp) > t$
(ECP4) $d(\mathcal{A}) + d(\mathcal{C}) > n$

The definition might seem abstract, however each of the conditions (ECP1)-(ECP4), will be used to prove the correctness of the ECP algorithm. For an overview of the primary roles of each of the conditions, we refer the reader to Table 4.1. Before moving on we will show our first example of a t-error correcting pair.

Example 4.14. Let $\mathcal{C}$ be a $[n,k,n-k+1]_q$ Reed-Solomon code and $t \leq \left\lfloor \frac{n-k}{2} \right\rfloor$. In addition let $\mathcal{A}$ be a Reed-Solomon code of dimension $t+1$, and $\mathcal{B}$ the dual code of a Reed-Solomon code of dimension $t+k$. If $\mathcal{A}$, $\mathcal{B}^\perp$ and $\mathcal{C}$ share the same evaluation points, say $P_1, P_2, \ldots, P_n$, then $(\mathcal{A},\mathcal{B})$ is a t-error correcting pair for $\mathcal{C}$. We start by showing condition (ECP1). Suppose $a \in \mathcal{A}$ and $c \in \mathcal{C}$, then $a = (F(P_1), F(P_2), \ldots, F(P_n))$ for some $F \in \mathbb{F}_q[X]_{<t+1}$ and $c = (g(P_1), g(P_2), \ldots, g(P_n))$ for some $G \in \mathbb{F}_q[X]_{<k}$. We see that:

$$a * c = (FG(p_1), FG(p_2), \ldots, FG(p_n)) \in \mathcal{B}^\perp$$

as $\deg(FG) < k+t$, since $\deg(F) < t+1$ and $\deg(G) < k$. The rest follows by applying Lemma 4.9. Condition (ECP2) is clearly meet, since $\dim_{\mathbb{F}_q}(\mathcal{A}) = t+1$ by our construction, and condition (ECP3) follows as $d(\mathcal{B}^\perp) = n - (t+k) + 1 \geq n - \frac{n-k}{2} - k + 1 = \frac{n-k}{2} + 1 > t$. Finally:

$$\begin{aligned} d(\mathcal{A}) + d(\mathcal{C}) &= (n - (t+1) + 1) + (n - k + 1) \\ &\geq 2n - k + 2 - \frac{n-k}{2} = \frac{2n + (n-k+1) + 3}{2} > n + \frac{3}{2} > n \end{aligned}$$

since $0 < d(\mathcal{C}) = n - k + 1$, which shows that (ECP4) is satisfied. □

We can now describe the ECP algorithm. We fix $y := c + e$ such that $c \in \mathcal{C}$ and $e \in \mathbb{F}_q^n$ with $\mathrm{wt}(e) \leq t$ for the rest of the section. Just like in the basic algorithm, the ECP algorithm consists of two steps: First we locate the errors by finding a subset $J \subseteq \{1,\ldots,n\}$ such that $\mathrm{supp}(e) \subseteq J$, secondly we apply Proposition 4.1, to find e, if possible.

One of the main ideas is to investigate the space:

$$\{a \in \mathcal{A} | a * e = 0\} = \mathcal{A}(\mathrm{supp}(e))$$

the equality follows as $a * e = 0$ if and only if $a_i = 0$ for all $i \in \mathrm{supp}(e)$. This vector space consists of vectors whose entries indexed by $\mathrm{supp}(e)$ are all 0, hence they can be used to locate the errors. Thus we want $\mathcal{A}(\mathrm{supp}(e))$ to contain at least one non-zero vector, which we can use to locate the errors.

Lemma 4.15. *Let $(\mathcal{A}, \mathcal{B})$ be a t-error correcting pair for $\mathcal{C}$ and $y = c + e$ with $c \in \mathcal{C}$ and $e \in \mathbb{F}_q^n$ with $\mathrm{wt}(e) \leq t$. Then:*

$$\mathcal{A}(\mathrm{supp}(e)) \neq \{0\}$$

Proof. Let $m = \dim_{\mathbb{F}_q}(\mathcal{A})$ and $a_1, a_2, \ldots, a_m \in \mathbb{F}_q^n$ form a basis of $\mathcal{A}$. Consider the vectors $a_1 * e, a_2 * e, \ldots, a_m * e$, then $\mathrm{wt}(a_i * e) \leq t$ as $\mathrm{wt}(e) \leq t$. Now since $\mathrm{supp}(a_i * e) \subseteq \mathrm{supp}(e)$ and $|\mathrm{supp}(e)| = \mathrm{wt}(e) \leq t < m$, by (ECP2). Using this information we can conclude that the vectors $a_1 * e, a_2 * e, \ldots, a_m * e$ are $\mathbb{F}_q$-linearly dependent, thus the equation:

$$\sum_{i=1}^{m} x_i(a_i * e) = 0$$

has no unique solution, where $x \in \mathbb{F}_q^m$, meaning multiple $a \in \mathcal{A}$ exists such that $a * e = 0$. ■

This is all quite theoretical, since e and in particular $\mathrm{supp}(e)$ is unknown in practical applications, we do not have any information about $A(\mathrm{supp}(e))$. Instead we consider the following vector space:

Definition 4.16. Let $(\mathcal{A}, \mathcal{B})$ be a t-error correcting pair for $\mathcal{C}$, and $y = c + e$, with $c \in \mathcal{C}$ and $e \in \mathbb{F}_q^n$ such that $\mathrm{wt}(e) \leq t$. Then we define:

$$M_{ECP} := \left\{ a \in \mathcal{A} \middle| a * y \in \mathcal{B}^{\perp} \right\}$$

The vector space M_{ECP} will play a similar role, in the ECP algorithm, to the role of the set $K_y(F)$ in the basic algorithm.

Theorem 4.17. *Let $(\mathcal{A}, \mathcal{B})$ be a t-error correcting pair for $\mathcal{C}$. Furthermore let $y = c + e$ with $c \in \mathcal{C}$ and $e \in \mathbb{F}_q^n$ with $\mathrm{wt}(e) \leq t$. Then:*

(i) $\mathcal{A}(\mathrm{supp}(e)) \subseteq M_{ECP} \subseteq \mathcal{A}$.

(ii) *If $d(\mathcal{B}^{\perp}) > t$, then $\mathcal{A}(\mathrm{supp}(e)) = M_{ECP}$.*

Proof. We start by proving Assertion (i), the inclusion $M_{ECP} \subseteq \mathcal{A}$ follows straight away from the definition of M_{ECP}. Next assume that $a \in \mathcal{A}(\mathrm{supp}(e))$, then for all $b \in \mathcal{B}$ we have:

$$\langle a * y, b \rangle \overset{(a)}{=} \langle a * c, b \rangle + \langle a * e, b \rangle \overset{(b)}{=} \langle a * b, c \rangle + \langle a * e, b \rangle \overset{(c)}{=} 0 \tag{4.3}$$

where (a) follows since $\langle \cdot, \cdot \rangle$ is linear in the first argument, (b) follows by Remark 4.8 and (c) as $\mathcal{A} * \mathcal{B} \subseteq \mathcal{C}^{\perp}$, by (ECP1) and $a * e = 0$. From Equation (4.3) we see that $a * y \in \mathcal{B}^{\perp}$ and hence $\mathcal{A}(\mathrm{supp}(e)) \subseteq M_{ECP}$.
Continuing with Assertion (ii). We only need to prove that $A(\mathrm{supp}(e)) \supseteq M_{ECP}$. Hence let $a \in M_{ECP}$, then $a * y = a * c + a * e$, so $a * e = a * y - a * c \in \mathcal{B}^{\perp}$, since $a * y \in \mathcal{B}^{\perp}$ by definition and $\mathcal{A} * \mathcal{C} \subseteq \mathcal{B}^{\perp}$ by Lemma 4.9 and (ECP1). Finally $\mathrm{wt}(a * e) \leq \mathrm{wt}(e) \leq t < d(\mathcal{B}^{\perp})$, by (ECP3), so $a * e = 0$, meaning $a \in \mathcal{A}(\mathrm{supp}(e))$. ■

Lemma 4.18. *Let $(\mathcal{A}, \mathcal{B})$ be a t-error correcting pair for the linear code $\mathcal{C}$, and $y = c + e$ with $c \in \mathcal{C}$ and $e \in \mathbb{F}_q^n$ such that $\mathrm{wt}(e) \leq t$. Then $|Z(M_{ECP})| \leq d(\mathcal{C})$.*

Proof. By Theorem 4.17(i) and Lemma 4.15 there exists a $a \in \mathcal{A}(\text{supp}(e)) \setminus \{0\} \subseteq M_{ECP}$. Since $Z(M_{ECP}) \subseteq Z(a)$ we get that:

$$|Z(M_{ECP})| \leq |Z(a)| = n - \text{wt}(a) \leq n - d(\mathcal{A}) \leq d(\mathcal{C})$$

by (ECP4). ■

This means that we can apply Proposition 4.1, to find the error e. This was the final piece of the puzzle, so we can now describe the ECP decoding algorithm.

Algorithm 4.19 Error Correcting Pairs Decoding Algorithm

procedure ECP DECODING(y: a received word with a maximum of t errors,
$(\mathcal{A}, \mathcal{B})$: a t-error correcting pair for $\mathcal{C}$,
H: a parity check matrix for $\mathcal{C}$)
$M_{ECP} \leftarrow \{a \in \mathcal{A} | a * y \in \mathcal{B}^{\perp}\}$
$I \leftarrow Z(M_{ECP})$
$S \leftarrow \{e \in \mathbb{F}_q^n | He = Hy \text{ and } e_i = 0 \text{ for all } i \in \{1, \ldots, n\} \setminus I\}$
if $|S| \neq 1$ **then**
return ?
else
return $y - e$ where e is the unique solution in S.

Remark 4.20. The time complexity of Algorithm 4.19, is also $O(n^{\omega})$ with $\omega \leq 3$. This is can be seen as follows: The set S is simply a solution set to a linear system, hence it can be computed in $O(n^{\omega})$. Additionally computing M_{ECP} can be done in $O(n^{\omega})$ assumming we have a generator matrices $G_{\mathcal{A}}$ and $G_{\mathcal{B}}$ of $\mathcal{A}$ and $\mathcal{B}$ respectively. Then M_{ECP} is nothing but the solution space to the equation:

$$G_{\mathcal{B}}(y^T * a^T) = 0 \tag{4.4}$$

Using $G'_{\mathcal{B}} := \begin{bmatrix} G'_{\mathcal{B}_{*,1}} y_1 & G'_{\mathcal{B}_{*,2}} y_2 & \cdot & G'_{\mathcal{B}_{*,n}} y_n \end{bmatrix}$ we may rewrite Equation (4.4) as $G'_{\mathcal{B}} a = 0$. Additionally we may substitue a with $x^T G_{\mathcal{A}}$ and solve for x instead, since $a \in \mathcal{A}$. Then we obtain

$$G'_{\mathcal{B}}(x^T G_{\mathcal{A}})^T = 0 \iff (G'_{\mathcal{B}} G_{\mathcal{A}}^T) x = 0$$

solving this equation for x we get a solution in parametric form. Say $x = v_0 + \sum_{i=1}^{m} c_i v_i$ with $v_i \in \mathbb{F}_q^k$ fixed and c_i arbitrary. Thus:

$$M_{ECP} = \left\{ \left(v_0^T + \sum_{i=1}^{m} c_i v_i^T \right) G_{\mathcal{A}} \middle| c_1, c_2, \ldots, c_m \in \mathbb{F}_q \right\} \tag{4.5}$$

TODO when is $a_i = 0$ for all $a \in M_{ECP}$.

Below in Table 4.1, the role of each ECP condition, is briefly explained, we note that there is a natural overlap, between the roles of some the conditions.

Condition	Role in the ECP algorithm
(ECP1)	Use to ensure $A(\text{supp}(e)) \subseteq M_{ECP}$ and that $\mathcal{A}(\text{supp}(e)) = M_{ECP}$.
(ECP2)	Ensures that $A(\text{supp}(e)) \neq \{0\}$ and $M_{ECP} \neq \{0\}$.
(ECP3)	Ensures that $A(\text{supp}(e)) = M_{ECP}$.
(ECP4)	Ensures that $\|Z(M_{ECP})\| \leq d(\mathcal{C})$, thus Proposition 4.1 can be applied.

Table 4.1: *The roles of each ECP condition.*

Finally to apply the theory of t-error correcting pairs to AG codes we a way to find a t-error correcting pair for a given AG code, luckily the next theorem provides exactly this. In addition it is a "natural" generalization of the approach taken in Example 4.14.

Theorem 4.21. *Let $\mathcal{X}$ be a regular absolutely projective algebraic curve of genus g over $\mathbb{F}_q$, and $\mathcal{C}_{D,G}$ be an AG code, constructed on $\mathcal{X}$, with $D = \sum_{i=1}^{n} P_i$. Let $t \leq \left\lfloor \frac{d^*-1}{2} - \frac{g}{2} \right\rfloor$ and let F be a divisor on $\mathcal{X}$ such that $\operatorname{supp}(F) \cap \operatorname{supp}(D) = \emptyset$ and $\deg(F) = t + g$ as well as $\deg(F+G) < n$. Then $(\mathcal{C}_{D,F}, \mathcal{C}^{\perp}_{D,F+G})$ forms a t-error correcting pair for $\mathcal{C}_{D,G}$.*

Proof. For the sake of simplicity we let $\mathcal{A} := \mathcal{C}_{D,F}$ and $\mathcal{B} := \mathcal{C}^{\perp}_{D,F+G}$ We start by showing that (ECP1) holds. Let $a \in \mathcal{C}_{D,F} = \mathcal{A}$ and $c \in \mathcal{C}_{D,G}$ then there exists some $f \in L(F)$ and $h \in L(G)$ such that $a = (f(P_1), f(P_2), \ldots, f(P_n))$ and $c = (h(P_1), h(P_2), \ldots, h(P_n)) \in L(G)$. using this information we see that:

$$a * c = (fh(P_1), fh(P_2), \ldots, fh(P_n)) \in \mathcal{C}_{D,F+G}$$

since $L(F)L(G) \subseteq L(F+G)$ by Remark 4.2. Thus:

$$\mathcal{A} * \mathcal{C}_{D,F} = \mathcal{C}_{D,F} * \mathcal{C}_{D,G} \subseteq C_{D,F+G} = \mathcal{B}^{\perp}$$

Continuing using $\deg(F) < \deg(F+G) < n$, since $\operatorname{supp}(F) \cap \operatorname{supp}(G) = \emptyset$, we see that:

$$\dim_{\mathbb{F}_q}(\mathcal{A}) \overset{(a)}{=} \ell(F) \overset{(b)}{\geq} \deg(F) - g + 1 = t + g - g + 1 = t + 1$$

where (a) follows by Theorem 3.2(iii) and (b) by the Riemann-Roch Theorem B.2. This shows that $\mathcal{A}$ satisfies (ECP2). Combining Theorem 3.2(ii) and the fact that $\deg(G+F) = \deg(G) + t + g$ yields (ECP3). This is seen as follows:

$$\begin{aligned} d(\mathcal{B}^{\perp}) \geq n - \deg(G+F) &= n - \deg(G) - t - g \\ &\overset{(c)}{\geq} n - \deg(G) - \frac{n - \deg(G) - g - 1}{2} - g \\ &= \frac{n - \deg(G) - g - 1}{2} \overset{(d)}{\geq} t \end{aligned}$$

where (c) and (d) follows as $t \leq \left\lfloor \frac{d^*-g-1}{2} \right\rfloor$ with $d^* = n - \deg(G)$. Finally we need to verify that (ECP4) holds. By applying Theorem 3.2(ii) we see that:

$$d(\mathcal{A}) + d(\mathcal{C}_{D,G}) > n - \deg(F) + n - \deg(G) = \deg(F+G) > n$$

by our original assumption. ■

4.2 Error Correcting Arrays

As we have seen previously both the basic algorithm described in Algorithm 4.4 and the error correcting pairs algorithm described in Algorithm 4.19, suffers under a genus penalty, meaning they can correct up to:

$$\left\lfloor \frac{d^* - 1}{2} - \frac{g}{2} \right\rfloor$$

where d^* is the designed minimum distance of $\mathcal{C}_L(\mathcal{X}, D, G)$ and g is the genus of $\mathcal{X}$. This isn't a problem when $g = 0$, which is the case when $\mathcal{X} = \mathbb{P}^1$ (atleast in the planar case.).

Hence we may apply those algorithms to Reed Solomon codes or even classical Goppa codes, which we shall see in Chapter 5, without being concerned. The algorithm described in this section deals with the case where $g \neq 0$, and will allow us to correct up to $\left\lfloor \frac{d^*-1}{2} \right\rfloor$ errors.

Definition 4.22. Let $A_u = \{\mathcal{A}_i\}_{i=1}^{u}$, $B_v = \{\mathcal{B}_j\}_{j=1}^{v}$ and $C_{w,l} := \{\mathcal{C}_r\}_{r=w}^{l}$ be sequences linear codes over $\mathbb{F}_q$ of length n. Then the triple $(A_u, B_v, C)_{w,l}$ is called an *array of codes* if the following conditions hold:

(ECA1) $\dim_{\mathbb{F}_q}(\mathcal{A}_i) = i$, $\dim_{\mathbb{F}_q}(\mathcal{B}_j) = j$ and $\dim_{\mathbb{F}_q}(\mathcal{C}_r) = n - r$.

(ECA2) The sequences A and B are increasing while C is decreasing, that is:

$$\mathcal{A}_i \subseteq \mathcal{A}_{i+1}, \mathcal{B}_j \subseteq \mathcal{B}_{j+1} \text{ and } \mathcal{C}_r \subseteq \mathcal{C}_{r-1}$$

(ECA3) For every pair (i, j) there exists at least one index r such that $\mathcal{A}_i * \mathcal{B}_j \subseteq \mathcal{C}_r^{\perp}$. We will denote the smallest such r by $r(i, j)$.

(ECA4) If $a \in \mathcal{A}_i \backslash \mathcal{A}_{i-1}$ and $b \in \mathcal{B}_j \backslash \mathcal{B}_{j-1}$ and $r(i, j) \geq w+1$, then $a*b \in \mathcal{C}_{r(i,j)}^{\perp} \backslash \mathcal{C}_{r(i,j)-1}^{\perp}$.

Remark 4.23. the function $r(i, j)$ is increasing (meaning $i \leq i'$ and $j \geq j'$ implies $r(i, j) \geq r(i', j')$). This follows from the fact that C is a decreasing sequence and hence the sequence $\left\{\mathcal{C}_r^{\perp}\right\}_{r=w}^{l}$ is increasing.

We will let

$$N_r = \{(i, j) | i \in \{1, 2, \ldots, u\}, j \in \{1, 2, \ldots, v\}, r(i, j) = r + 1\}$$

and $n_r = |N_r|$ as well as:

$$d_r = \min\left\{n_{r'} | r \leq r' < l\right\} \cup \{d(\mathcal{C}_l)\}$$

We note that $d_r \leq d_{r+1}$, since $\min\{|N_{r'}| | r \leq r' < l\} \leq \min\{|N_{r'}| | r + 1 \leq r' < l\}$.

Theorem 4.24. *For any array of codes $(A_u, B_v, C_{w,l})$ we have $d_r \geq d(\mathcal{C}_r)$ for $r = w, w + 1, \ldots, l$.*

To prove the Theorem, we will show that $d_l \geq d(\mathcal{C}_l)$ and show that $d_{r+1} \geq d(\mathcal{C}_{r+1})$ implies that $d_r \geq d(\mathcal{C}_r)$. This resembles mathematical induction, except we are going in the oppisite direction, however this is sufficient, since our sequence $C_{w,l}$ only consists of $l - w$ elements.

Proof. We start by proving that $d_l \geq d(\mathcal{C}_l)$, this follows straight from the definition of d_l. Next assume that $d_{r+1} \geq d(\mathcal{C}_{r+1})$. Let $y \in \mathcal{C}_r \setminus \{0\}$ we will show that $\mathrm{wt}(y) \leq d_r$, meaning $d(\mathcal{C}_r) \leq d_r$. If $y \in \mathcal{C}_{r+1} \setminus \{0\}$ then $\mathrm{wt}(y) \geq d_{r+1} \geq d_r$, where the first inequality follows by our hypothesis and the second from the definition of d_r. On the otherhand suppose that $y \in \mathcal{C}_r \setminus \mathcal{C}_{r+1}$. Let $a_1, a_2, \ldots, a_i$ and $b_1, b_2, \ldots, b_j$ be bases for $\mathcal{A}_i$ and $\mathcal{B}_i$ respectively, we note that such bases exist by (ECA1) and (ECA2). Then $a_i \in \mathcal{A}_i \setminus \mathcal{A}_{i-1}$ and $b_j \in \mathcal{B}_j \setminus \mathcal{B}_{j-1}$, since $\dim_{\mathbb{F}_q}(\mathcal{A}_i) = i$ and $\dim_{\mathbb{F}_q}(\mathcal{B}_j) = j$ by (ECA1). Hence $a_i * b_j \in \mathcal{C}_{r(i,j)}^{\perp} \setminus \mathcal{C}_{r(i,j)-1}^{\perp}$, by (ECA4). Consider the matrix:

$$S = \begin{bmatrix} \langle y, a_1 * b_1 \rangle & \langle y, a_1 * b_2 \rangle & \cdots & \langle y, a_1 * b_v \rangle \\ \langle y, a_2 * b_1 \rangle & \langle y, a_2 * b_2 \rangle & \cdots & \langle y, a_2 * b_v \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle y, a_u * b_1 \rangle & \langle y, a_u * b_2 \rangle & \cdots & \langle y, a_u * b_v \rangle \end{bmatrix}$$

Where $\langle \cdot, \cdot \rangle$, denotes the usual dot product. If $r(i,j) \leq r$, then $S_{i,j} = 0$ by (ECA3), since $y \in \mathcal{C}_r$ and $a_i * b_j \in \mathcal{C}_r^\perp$. If $r(i,j) = r+1$, then $S_{i,j} \neq 0$. Since $\mathcal{C}_{r+1}^\perp = \mathcal{C}_r^\perp \oplus \operatorname{span}_{\mathbb{F}_q}\{a_i * b_j\}$ as $a_i * b_j \in \mathcal{C}_{r+1}^\perp \setminus \mathcal{C}_r^\perp$ and $\dim_{\mathbb{F}_q}(\mathcal{C}_r) = \dim_{\mathbb{F}_q}(\mathcal{C}_{r+1}) + 1$ by (ECP1) meaning:

$$\dim_{\mathbb{F}_q}(\mathcal{C}_{r+1}^\perp) = n - \dim_{\mathbb{F}_q}(\mathcal{C}_{r+1}) = n - (\dim_{\mathbb{F}_q}(\mathcal{C}_r) - 1) = \dim_{\mathbb{F}_q}(\mathcal{C}_r^\perp) + 1$$

Thus S has a row echelon form with pivots atleast at all entries $S_{i,j}$ such that $r(i,j) = r+1$. Hence:

$$\operatorname{rank}(S) \geq |N_r| = n_r \tag{4.6}$$

Additionally we may write S as:

$$S = \begin{bmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_u^T \end{bmatrix} \operatorname{diag}(y) \begin{bmatrix} b_1 & b_2 & \dots & b_v \end{bmatrix}$$

where $\operatorname{diag}(y)$ is $n \times n$ the diagonal matrix with the entries of y in the diagonal. Thus

$$\operatorname{rank}(S) \leq \min\{u, \operatorname{rank}(\operatorname{diag}(y)), v\} \leq \operatorname{rank}(\operatorname{diag}(y)) = \operatorname{wt}(y) \tag{4.7}$$

since $\operatorname{rank}(AB) \leq \operatorname{rank}(A)\operatorname{rank}(B)$ for all matrices A and B. Combining inequalities (4.6) and (4.7) we get $n_r \leq \operatorname{rank}(S) \leq \operatorname{wt}(y)$ the rest follows as $d_r \leq n_r$. ■

Definition 4.25. Let $(A_u, B_v, C_{w,l})$ be an array of codes and $\mathcal{C}$ be a $\mathbb{F}_q^n$ linear code. Then (A, B, C) is called a t-error correcting array if $\mathcal{C}_w = \mathcal{C}$ and $t \leq \left\lfloor \frac{d_w - 1}{2} \right\rfloor$ and either:

(i) $C_l = \{0\}$

(ii) or there exists i, j such that (A_i, B_j) is a t-error correcting pair for $\mathcal{C}_{r(i,j)}$.

We will fix some notation for the rest of the section. Namely let $(A_u, B_v, C_{w,l})$ be a t-error correcting array for $\mathcal{C}$ and $y = c + e$ with $c \in \mathcal{C}$ and $e \in \mathbb{F}_q^n$ with $\operatorname{wt}(e) \leq t$. Fix the matrix

$$S := \begin{bmatrix} \langle e, a_1 * b_1 \rangle & \langle e, a_1 * b_2 \rangle & \cdots & \langle e, a_1 * b_u \rangle \\ \langle e, a_2 * b_1 \rangle & \langle e, a_2 * b_2 \rangle & \cdots & \langle e, a_2 * b_u \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle e, a_v * b_1 \rangle & \langle e, a_v * b_2 \rangle & \cdots & \langle e, a_v * b_u \rangle \end{bmatrix}$$

where $a_1, a_2, \dots, a_i$ and $b_1, b_2, \dots, b_j$ forms a basis of $\mathcal{A}_i$ and $\mathcal{B}_j$ respectively. Additionally we will let:

$$S|_{(i',j')} = \begin{bmatrix} S_{1,1} & S_{1,2} & \cdots & S_{1,j'} \\ S_{2,1} & S_{2,2} & \cdots & S_{2,j'} \\ \vdots & \vdots & \ddots & \vdots \\ S_{i',1} & S_{i',2} & \cdots & S_{i',j'} \end{bmatrix}$$

Definition 4.26. Consider the conditions

(D1) $\operatorname{rank}(S|_{(i-1,j-1)}) = \operatorname{rank}(S|_{(i-1,j)}) = \operatorname{rank}(S|_{(i,j-1)})$.

(D2) $\operatorname{rank}(S|_{(i-1,j-1)}) = \operatorname{rank}(S|_{(i,j)})$.

The pair (i,j) is called a *discrepancy* if (D1) holds but (D2) does not.

Clearly (D2) implies (D1), afterall $\text{rank}(S|_{(i-1,j-1)}) = \text{rank}(S|_{(i,j)})$ if and only if the last row and or column in $S|_{(i,j)}$ is a $\mathbb{F}_q$-linear combination of the others. The converse is not that obvious:

Lemma 4.27. *Suppose (D1) holds for the pair (i,j) then there exists a unique value of $S_{i,j}$ such that (D2) holds.*

Proof. Since (D1) holds, there exists $c \in \mathbb{F}_q^j$ and $k \in \mathbb{F}_q^j$, with $\text{wt}(c) = \text{wt}(k) = \text{rank}(S|_{(i-1,j-1)})$ such that $c_i \neq k_j$ and:

$$\sum_{l=1}^{i} c_l \begin{bmatrix} S_{l,1} & S_{l,2} & \cdots & S_{l,j-1} \end{bmatrix}^T = \sum_{l=1}^{j} k_l \begin{bmatrix} S_{1,l} \\ S_{2,l} \\ \vdots \\ S_{i-1,l} \end{bmatrix} = 0$$

Hence (D2) holds if and only if:

$$\sum_{l=1}^{i-1} c_l S_{l,j} + c_i S_{i,j} = \sum_{l=1}^{j-1} k_l S_{i,l} + k_j S_{i,j} = 0$$

Solving for $S_{i,j}$ using the fact that $c_i \neq k_j$, gives:

$$S_{i,j} = \frac{\sum_{l=1}^{j-1} k_l S_{i,l} - \sum_{l=1}^{i-1} c_l S_{l,j}}{(c_i - k_j)}$$

Finally we note that if $S_{i,j} \neq \frac{\sum_{l=1}^{j-1} k_l S_{i,l} - \sum_{l=1}^{i-1} c_l S_{l,j}}{(c_i - k_j)}$, then $\text{rank}(S|_{(i,j)}) \neq \text{rank}(S|_{i-1,j-1})$ since the row i and or column j is not in the span of the first $i-1$ rows or $j-1$ columns respectively. ■

Remark 4.28. Condition (D1) holds for the pair (i,j) if and only if (D2) holds for all (i',j) with $1 \leq i' < i$ and all (i,j') with $1 \leq j' < j$. needs a proof. Thus there is at most one discrepancy in each row and column of S.

Lemma 4.29. *Let M denote the number of discrepancies of S, then $M \leq \text{rank}(S)$*

Proof. Assume for the sake of contradiction that $M > \text{rank}(S)$. Then there exists M 1 (i_k, j_k) for $k = 1, 2, \ldots, M$, we will assume that these are in lexicographic order, additionally we note that each row and or column can have at most one discrepancy, by Remark **??**. Since condition (D2) does not hold for (i_k, j_k) we see that:

$$\text{rank}(S|_{(i_{k-1}, j_{k-1})}) < \text{rank}(S|_{(i_k, j_k)})$$

Since $\text{rank}(S|_{(i_k,j_k)})$ is the maximum number of linearly independent columns / rows of $S|_{(i_k,j_k)}$. Hence $\left\{\text{rank}(S|_{(i_k,j_k)})\right\}_{k=1}^{M}$ is a strictly increasing sequence, thus $\text{rank}(S) \geq \text{rank}(S|_{i_M,j_M}) \geq M$, since $S|_{i_M,j_M}$ is a submatrix. ■

Definition 4.30. Let $y_r \in \mathcal{C}_r + e$, the pair (i,j) is called a *r-candidate* if $r(i,j) = r+1$ and (D1) holds. A candidate (i,j) is called *true* if (D2) holds and *false* otherwise.

Proposition 4.31. *Let $y \in \mathcal{C}_r + e$ and T denote the number of true r-candidates and F denote the number of false r-candidates. Then:*

$$F < T$$

Proof. If (i,j) is a discrepancy such that $r(i,j) \leq r$, then (i,j) is called a known discrepancy, otherwise if $r(i,j) > r$, then (i,j) is called an unknown discrepancy. Let K denote the number of known discrepancies. Let (i,j) be a false r-candidate then $r(i,j) = r+1 > r$, hence every false r-candidate is a unknown discrepancy, thus:

$$K + F \leq M \leq \operatorname{rank}(S) \leq t \tag{4.8}$$

where M denotes the total number of discrepancies of S, two final inequalities follows by Lemma 4.29 and the proof of Theorem 4.24 respectively. Consider the pair $(i,j) \in N_r$ if (i,j) is not a candidate there exists a known discrepency in the same row and or column. On the otherhand if (i,j) is a candidate, then there exists no known discrepancy in the same row nor column, by Lemma **??**. Hence:

$$n_r \leq T + F + 2K \tag{4.9}$$

Finally combining Inequalities (4.8) and (4.9) we see:

$$n_r \leq T + F + 2K \leq T + F + (2t - 2F) = T - F + 2t < T - F + n_r$$

where the last inequality follows as $2t < d_w \leq d_r < n_r$, since $\{d_r\}_{r=w}^{l}$ is an increasing sequence, and $d_r = \min\{n_{r'} | r \leq r' < l\} \cup \{d(\mathcal{C}_l)\}$. Thus $0 < T - F$ meaning $F < T$. ■

Theorem 4.32. *If $\mathcal{C}$ has a t error correcting array (A,B,C), then it has a t error correcting decoding algorithm with a time complexity of $O(n^3)$.*

Proof. Suppose $y_w = c_w + e$ with $c_w \in \mathcal{C} = \mathcal{C}_w$, $e \in \mathbb{F}_q^n$ and $\operatorname{wt}(e) \leq t \leq \lfloor \frac{d_w - 1}{2} \rfloor$. We will show that for each $y_r \in e + \mathcal{C}_r$ we can construct a $y_{r+1} \in e + \mathcal{C}_{r+1}$. The fundemental idea will be to associate a $\lambda \in \mathbb{F}_q$ with each candidate such that all true candidates have the same λ. Hence since $F < T$ by Proposition 4.31 where F denotes the number of false candidates and T denotes the number of false candidates, this will allow us to identify the false and true candidates, by majority. Hence suppose that we have $y_r \in e + \mathcal{C}_r$. For every candidate (i,j) there exists a unique $S'_{i,j} \in \mathbb{F}_q$ such that (D2) holds if and only if $S_{i,j} = S'_{i,j}$, by Lemma 4.27. We may compute $S'_{i,j}$, as seen in the proof of Lemma 4.27, using the known entries of S, meaning the entries $S_{i,j}$ with $r(i,j) < r$. Additionally we note that $\dim_{\mathbb{F}_q}(\mathcal{C}_r) = \dim_{\mathbb{F}_q}(\mathcal{C}_{r+1}) + 1$ by (ECA1) and $\mathcal{C}_{r+1} \subseteq \mathcal{C}_r$, hence there exists a $c_r \in \mathbb{F}_q^n$ such that $\mathcal{C}_r = \operatorname{span}_{\mathbb{F}_q}\{c_r\} \oplus \mathcal{C}_{r+1}$. Hence there exists a unique $\lambda_r \in \mathbb{F}_q$ such that $y_{r+1} := y_r + \lambda_r c_r \in e + \mathcal{C}_{r+1}$ as $y_r = c'_r + e$ with $c'_r \in \mathcal{C}_r$ and every element in $\mathcal{C}_{r+1}$ can be written uniquely as $c'_r + \lambda c_r$ for some λ.

If $r(i,j) = r+1$ then $\mathcal{C}_{r+1}^{\perp} = \mathcal{C}_r \oplus \operatorname{span}_{\mathbb{F}_q}\{a_i * b_i\}$ by (ECA4) and $\dim_{\mathbb{F}_q}(\mathcal{C}_{r+1}^{\perp}) = \dim_{\mathbb{F}_q}(\mathcal{C}_r^{\perp}) + 1$. We note that since $c_r \in \mathcal{C}_r$ we see that $\langle c_r, a_i * b_i \rangle \neq 0$ by (ECP4). Thus:

$$\langle y_{r+1}, a_i * b_i \rangle = \langle y_r, a_i * b_i \rangle + \lambda_r \langle c_r, a_i * b_j \rangle = S_{i,j}$$

as $S_{i,j} = \langle e, a_i * b_j \rangle = \langle c_{r+1}, a_i * b_j \rangle + \langle e, a_i * b_j \rangle = \langle y_{r+1}, a_i * b_j \rangle$ as $a_i * b_j \in \mathcal{C}_{r+1}^{\perp}$, we note that such a $c_{r+1} \in \mathcal{C}_{r+1}$ exists as $y_{r+1} \in e + \mathcal{C}_{r+1}$. Thus for every candidate (i,j) we let

$$\lambda_{i,j} := \frac{S'_{i,j} - \langle y_r, a_i * b_j \rangle}{\langle c_r, a_i * b_j \rangle}$$

hence if $S'_{i,j} = S_{i,j}$ meaning (i,j) is a true candidate, we see that $\lambda_{i,j} = \lambda_r$. Hence the $\lambda_{i,j}$ which occurs most often is equal to λ_r. In this way we find $y_{r+1} \in e + \mathcal{C}_{r+1}$ and the syndromes $S_{i,j}$ such that $r(i,j) \leq r+1$. We continue this process until we reach $\mathcal{C}_l = \{0\}$, in which case we are done as $y_l = e$, or a $\mathcal{C}_r$ with a t-error correcting pair $(\mathcal{A}_i, \mathcal{B}_j)$ with $r = r(i,j)$ and $t \leq \frac{d_w - 1}{2}$ in which case we apply the error correcting pairs algorithm. ■

TODO mangler et resultat som giver os at der findes ECA for AG koder.

5 Classical Goppa Codes

Next we introduce classical Goppa codes. The original proposal of McElice used these codes in its construction. In addition they remain as one of the few classes codes proposed for with no publicly known structural attacks, at least if they are chosen correctly.

Theorem 5.1. *Let $f \in \mathbb{F}_q[X]$ and $\alpha \in \mathbb{F}_q$ such that $f(\alpha) \neq 0$, then the inverse of $(X-\alpha)$ exists in the quotient ring $\frac{\mathbb{F}_q[X]}{\langle f \rangle}$, and may be computed as $-\left(\frac{f(X)-f(\alpha)}{X-\alpha}\right) f(\alpha)^{-1}$.*

Proof. Follows from the fact that:

$$\begin{aligned} -\left(\frac{f(X)-f(\alpha)}{X-\alpha}\right) f(\alpha)^{-1}(X-\alpha) &= -f(\alpha)^{-1}(f(X)-f(\alpha)) \\ &= -f(\alpha)^{-1}f(X)+1 \equiv 1 \mod f \end{aligned}$$

Which is equivalent with $-\left(\frac{f(X)-f(\alpha)}{X-\alpha}\right) f(\alpha)^{-1}(X-\alpha) = 1 \in \frac{\mathbb{F}_q[X]}{\langle f \rangle}$. ∎

Definition 5.2. Let $x \in \mathbb{F}_q^n$ and $f \in \mathbb{F}_q[X]$ such that $f(x_i) \neq 0$ for $i = 1, 2, \ldots, n$ and $\mathbb{F}_{q_0} \subseteq \mathbb{F}_q$ be a subfield, then the *classical Goppa code* associated with $(x, f, \mathbb{F}_{q_0})$ is defined as:

$$\begin{aligned} \Gamma(x, f, \mathbb{F}_{q_0}) : &= \left\{ c \in \mathbb{F}_{q_0}^n \middle| \sum_{i=1}^{n} \frac{c_i}{X-x_i} \equiv 0 \mod f \right\} \\ &= \left\{ c \in \mathbb{F}_{q_0}^n \middle| \sum_{i=1}^{n} \frac{c_i}{X-x_i} = 0 \in \frac{\mathbb{F}_q[X]}{\langle f \rangle} \right\} \end{aligned}$$

If f is irreducible, then $\Gamma(x, f, \mathbb{F}_{q_0})$ is also called irreducible.

Remark 5.3. If f is irreducible and $\deg(f) = l$, then $\mathbb{F}_q[X]/\langle f \rangle$ is a finite field with q^l elements.

We note that $\Gamma(x, f, \mathbb{F}_{q_0})$ does indeed form a linear subspace of $\mathbb{F}_{q_0}^n$ since: If $c, c' \in \Gamma(x, f, \mathbb{F}_{q_0})$, we will show that this implies that $c + c' = \left[c_1 + c'_1, c_2 + c'_2, \ldots, c_n + c'_n\right] \in \Gamma(x, f, \mathbb{F}_{q_0})$. This follows since:

$$\sum_{i=1}^{n} \frac{c_i + c'_i}{X-x_i} = \sum_{i=1}^{n} \frac{c_i}{X-x_i} + \sum_{i=1}^{n} \frac{c'_i}{X-x_i} \equiv 0 \mod f$$

Additionally if $c \in \Gamma(x, f, \mathbb{F}_{q_0})$, then $kc \in \Gamma(x, f, \mathbb{F}_{q_0})$ for all $k \in \mathbb{F}_{q_0}$, since:

$$\sum_{i=1}^{n} \frac{kc_i}{X-x_i} = k \sum_{i=1}^{n} \frac{c_i}{X-x_i} \equiv 0 \mod f$$

From Theorem 5.1 and Definition 5.2 it follows that $c = (c_1, c_2, \ldots, c_n) \in \Gamma(x, f, \mathbb{F}_{q_0})$ if and only if

$$-\sum_{i=1}^{n} c_i \frac{1}{f(x_i)} \frac{f(X) - f(x_i)}{X - x_i} = 0 \in \frac{\mathbb{F}_q[X]}{\langle f \rangle} \tag{5.1}$$

additionally if $f = \sum_{j=0}^{\deg(f)} a_j X^j$, then:

$$\begin{aligned} -\frac{1}{f(x_i)} \frac{f(X) - f(x_i)}{X - x_i} &= -\frac{1}{f(x_i)} \frac{\sum_{j=0}^{\deg(f)} a_j (X^j - x_i^j)}{X - x_i} \\ &\overset{(a)}{=} -\frac{1}{f(x_i)} \sum_{j=1}^{\deg(f)} a_j \sum_{k=0}^{j-1} X^k x_i^{j-1-k} \\ &\overset{(b)}{=} -\frac{1}{f(x_i)} \sum_{k=0}^{\deg(f)-1} X^k \left(\sum_{j=k+1}^{\deg(f)} a_j x_i^{j-1-k} \right) \end{aligned}$$

where (a) follows by polynomial division and (b) from interchanging the sums. Thus:

$$\sum_{i=1}^{n} \frac{c_i}{X - x_i} = -\sum_{i=1}^{n} \frac{c_i}{f(x_i)} \sum_{k=0}^{\deg(f)-1} X^k \sum_{j=k+1}^{\deg(f)} a_j x_i^{j-1-k} \tag{5.2}$$

Equation (5.2) must equal 0 since $c \in \Gamma(x, f, \mathbb{F}_{q_0})$ which is the case if and only if the coefficients of each X^k is zero, which yields the following proposition:

Proposition 5.4. *Let $x \in \mathbb{F}_q^n$ and $f \in \mathbb{F}_q[X]$ with $l := \deg(f)$, then the classical gopppa code $\Gamma(x, f, \mathbb{F}_{q_0})$ has parity check matrix:*

$$H = \begin{bmatrix} f(x_1)^{-1} a_l & f(x_2)^{-1} a_l & \cdots & f(x_n)^{-1} a_l \\ f(x_1)^{-1}(a_l + a_{l-1} x_1) & f(x_2)^{-1}(a_l + a_{l-1} x_2) & \cdots & f(x_n)^{-1}(a_l + a_{l-1} x_n) \\ \vdots & \vdots & \ddots & \vdots \\ f(x_1)^{-1} \sum_{j=1}^{l} a_j x_1^{j-1} & f(x_2)^{-1} \sum_{j=1}^{l} a_j x_2^{j-1} & \cdots & f(x_n)^{-1} \sum_{j=1}^{l} a_j x_n^{j-1} \end{bmatrix}$$

over $\mathbb{F}_q$.

The following Corollary uses some elementary results from Galois theory, namely that if $\mathbb{F}_{q_0}$ is a subfield of the finite field $\mathbb{F}_q$, then there exists a $\mathbb{F}_{q_0}$-basis of $\mathbb{F}_q$. We will denote the length of such a basis by $[\mathbb{F}_q : \mathbb{F}_{q_0}]$ an call it the *degree* of the field extension $\mathbb{F}_q / \mathbb{F}_{q_0}$. Could be proven in an appendix.

Corollary 5.5. *Let $\Gamma(x, f, \mathbb{F}_{q_0})$ be a classical goppa code with $x \in \mathbb{F}_q$ and $f \in \mathbb{F}_q[X]$ then:*

$$\dim_{\mathbb{F}_{q_0}}(\Gamma(x, f, \mathbb{F}_{q_0})) \geq n - m \deg(f)$$

where $m = [\mathbb{F}_q : \mathbb{F}_{q_0}]$.

Proof. Fixing a $\mathbb{F}_{q_0}$-basis of $\mathbb{F}_q$ say $b_1, b_2, \ldots, b_m \in \mathbb{F}_q$, each entry in H can be viewed as a vector in $\mathbb{F}_{q_0}^m$, by the vector space isomorphism:

$$\mathbb{F}_q \ni x = x_1 b_1 + x_2 b_2 + \cdots + x_m b_m \mapsto \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix}^T \in \mathbb{F}_{q_0}^m$$

Thus H from Proposition 5.4 can be viewed as a $\mathbb{F}_{q_0}$-matrix H' with dimensions $m \deg(f) \times n$. Since H' has mt rows, we see that $\operatorname{rank}(H') \leq m \deg(f)$, hence

$$\dim_{\mathbb{F}_{q_0}}(\Gamma(x, f, \mathbb{F}_{q_0})) = \dim_{\mathbb{F}_{q_0}}(\operatorname{null}(H')) = n - \operatorname{rank}(H') \geq n - m \deg(f) \qquad \blacksquare$$

Example 5.6. Let $\mathbb{F}_4$ be the finite field with the four elements $0, 1, \alpha, 1+\alpha$ where $\alpha^2 = 1+\alpha$ and $x + x = 0$ for all $x \in \mathbb{F}_4$. We will let $f(X) = \alpha X^2 + X + (1+\alpha) \in \mathbb{F}_4[X]$ and $x = \begin{bmatrix} 0 & \alpha & 1+\alpha \end{bmatrix}$. We see that $f(x_1) = 1+\alpha$, $f(x_2) = \alpha$ while $f(x_3) = 1$. Next we will consider the Goppa code $\Gamma(x, f, \mathbb{F}_4)$. Applying proposition 5.4 to get a parity check matrix H, for $\Gamma(x, f, \mathbb{F}_4)$ we get:

$$H = \begin{bmatrix} 0 & 0 & 0 \\ (1+\alpha) & 1 & \alpha \\ (1+\alpha) & 0 & 1 \end{bmatrix}$$

Next we solve the linear system $Hc = 0$ to find the codewords of $\Gamma(x, f, \mathbb{F}_4)$, we start by noting that the augmented matrix $\begin{bmatrix} H|0 \end{bmatrix}$ is row equivilent with:

$$\begin{bmatrix} 1 & 0 & \alpha & 0 \\ 0 & 1 & (1+\alpha) & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Thus:

$$c = c_3 \begin{bmatrix} \alpha \\ (1+\alpha) \\ 1 \end{bmatrix} \iff c \in \Gamma(x, f, \mathbb{F}_4)$$

for all $c_3 \in \mathbb{F}_4$. Hence $\Gamma(x, f, \mathbb{F}_4) = \{(0,0,0), (\alpha, 1+\alpha, 1), (1+\alpha, 1, \alpha), (1, \alpha, 1+\alpha)\}$. Furthermore we note that $\Gamma(x, f, \mathbb{F}_4)$ is a $[3, 1, 3]_4$ code, meaning $\Gamma(x, f, \mathbb{F}_4)$ MDS code. □

Finally we reach the main theorem of this chapter.

Theorem 5.7. *Let $\Gamma(x, f, \mathbb{F}_q)$ be a classical Goppa code, $P_i = (x_i : 1)$ for $i = 1, 2, \ldots, n$ and $D = \sum_{i=1}^{n} P_i$. Then:*

$$\Gamma(x, f, \mathbb{F}_q) = \mathcal{C}_\Omega(\mathbb{P}^1, D, (f)_0 - P_\infty) = \mathcal{C}_L(\mathbb{P}^1, D, (\omega) + D - (f_0) + P_\infty) \tag{5.3}$$

with $\omega := \frac{dh}{h}$ where $h(x) := \prod_{i=1}^{n}(x - x_i)$.

Proof. The last equality in Equation (5.3) follows by Theorem 3.16. Hence it will be sufficient to show the first equality. We start by showing that $\Gamma(x, f, \mathbb{F}_q) \subseteq \mathcal{C}_\Omega(\mathbb{P}^1, D, (f)_0 - P_\infty)$. Hence let $c \in \Gamma(x, f, \mathbb{F}_q)$ and

$$\omega_c := \left(\sum_{i=1}^{n} \frac{c_i}{x - x_i} \right) dx$$

We note that x is a local parameter at all of the P_i's. By the definition of $\Gamma(x, f, \mathbb{F}_q)$ we see that the ω_c vanishes at all $P \in \operatorname{supp}((f)_0)$, meaning $v_P(\omega_c) > 0$, afterall $\sum_{i=1}^{n} \frac{c_i}{x - x_i} \equiv 0 \mod f$. In addition we note that $v_{P_i}(\omega_c) = -1$ and ω_c is regular at any other point $\mathbb{P}^1 \setminus \{P_\infty\}$. Finally we compute $v_{P_\infty}(\omega_c)$ by substituting x with $1/u$, we see that:

$$\omega_c \overset{(a)}{=} \sum_{i=1}^{n} \frac{-c_i \frac{du}{u^2}}{\frac{1}{u} - x_i} = -\sum_{i=1}^{n} \frac{c_i du}{u(1 - x_i u)} \tag{5.4}$$

where (a) follows since Remark 3.4 implies:

$$0 = D\left(\frac{1}{u} u\right) = D(u)\frac{1}{u} + uD\left(\frac{1}{u}\right)$$

which in turn impleis that $D\left(\frac{1}{u}\right) = -\frac{1}{u^2}D(u)$ for all derivations $D : \mathbb{F}(\mathcal{X}) \to \mathbb{F}(\mathcal{X})$. Thus by Equation (5.4) $v_{P_\infty}(\omega_c) \geq -1$ and hence $\omega_c \in \Omega((f)_0 - P_\infty - D)$ meaning $\Gamma(x, f, \mathbb{F}_q) \subseteq \mathcal{C}_\Omega(\mathbb{P}^1, D, (f)_0 - P_\infty)$. Conversely, given $\omega \in \Omega((f)_0 - P_\infty - D)$ we want to show that the codeword $c = (\mathrm{Res}_{P_1}(\omega), \mathrm{Res}_{P_2}(\omega), \ldots, \mathrm{Res}_{P_n}(\omega))$ is in $\Gamma(x, f, \mathbb{F}_q)$. Consider the differential:

$$\eta = \sum_{i=1}^{n} \frac{\mathrm{Res}_{P_i}(\omega)dx}{x - x_i}$$

Using the same argument as previously we see that the poles of η are contained in $\{P_1, P_2, \ldots, P_n, P_\infty\}$. We also note that $\mathrm{Res}_{P_i}(\eta) = \mathrm{Res}_{P_i}(\omega)$ for $i = 1, 2, \ldots, n$. Hence by the Residue Theorem 3.19 we see that $\mathrm{Res}_{P_\infty}(\eta) = \mathrm{Res}_{P_\infty}(\omega)$ since the residues of differentials at regular points are 0. Thus the differential $\eta - \omega$ has no poles on $\mathbb{P}^1$, all the residues are zero afterall. However the degree of a non-zero canonical divisor is $2g - 2 = -2$ by Proposition 3.12, as the genus of $\mathbb{P}^1$ is zero. Thus $\eta - \omega = 0$ meaning $\eta = \omega$ and thus $\sum_{i=1}^{n} \frac{\mathrm{Res}_{P_i}(\omega)}{x - x_i}$ vanishes on $(f)_0$, since $\eta = \omega \in \Omega((f)_0 - P_\infty - D)$. Moreover $\sum_{i=1}^{n} \frac{\mathrm{Res}_{P_i}(\omega)}{x - x_i}$ vanishing on $(f)_0$ is equivilent with:

$$\sum_{i=1}^{n} \frac{\mathrm{Res}_{P_i}(\omega)}{x - x_i} \equiv 0 \mod f$$

meaning $c \in \Gamma(x, f, \mathbb{F}_q)$. Since c was chosen arbitrarily, this shows that $\mathcal{C}_\Omega(\mathbb{P}^1, D, (f)_0 - P_\infty) \subseteq \Gamma(x, f, \mathbb{F}_q)$ concluding the proof. ■

Since Theorem 5.7 shows that classical Goppa codes are in fact AG codes it makes sense to speak about their designed minimum distance. This leads us to the following corollary:

Corollary 5.8. *If $f \in \mathbb{F}_q[X]$ splits into linear factors over $\mathbb{F}_q$, then:*

$$d^*\left(\Gamma(f, x, \mathbb{F}_q)\right) = \deg(f) + 1$$

Proof. From Corollary 3.17 it follows that the designed minimum distance of $\Gamma(f, x, \mathbb{F}_q)$ is $\deg((f)_0 - P_\infty) = \deg(f) - 1 + 2 = \deg(f) + 1$ ■

5.1 Subfield Subcodes

The following section is concerned with what happends if we only consider the codewords, of a linear code over $\mathbb{F}_q$, with entries in a subfield of $\mathbb{F}_q$. We start by motivating the concept by the following proposition:

Proposition 5.9. *Let $x \in \mathbb{F}_q^n$ and $f \in \mathbb{F}_q[X]$, such that $f(x_i) \neq 0$ for $i = 1, 2, \ldots, n$. Suppose $\mathbb{F}_{q_0}$ is a subfield of $\mathbb{F}_q$ then:*

$$\Gamma(x, f, \mathbb{F}_{q_0}) = \Gamma(x, f, \mathbb{F}_q) \cap \mathbb{F}_{q_0}^n$$

Proof. Follows straight from the definition, namely:

$$\begin{aligned}\Gamma(x, f, \mathbb{F}_{q_0}) &= \left\{ c \in \mathbb{F}_{q_0}^n \middle| \sum_{i=1}^{n} \frac{c_i}{X - x_i} \equiv 0 \mod f \right\} \\ &= \left\{ c \in \mathbb{F}_q^n \middle| \sum_{i=1}^{n} \frac{c_i}{X - x_i} \equiv 0 \mod f \right\} \cap \mathbb{F}_{q_0}^n = \Gamma(x, f, \mathbb{F}_q) \cap \mathbb{F}_{q_0}^n\end{aligned}$$ ■

We generalize the concept described in Proposition 5.9, with the following definition.

Definition 5.10. Let $\mathcal{C}$ be a $\mathbb{F}_q$ linear code and $\mathbb{F}_{q_0}$ be a subfield of $\mathbb{F}_q$, then the $\mathbb{F}_{q_0}$ linear code:

$$\mathcal{C}|_{\mathbb{F}_{q_0}} := \mathcal{C} \cap \mathbb{F}_{q_0}^n$$

is called a *subfield subcode* of $\mathcal{C}$.

Let $c_1, c_2 \in \mathcal{C}|_{\mathbb{F}_{q_0}}$, then $c_1 + c_2 \in \mathcal{C}|_{\mathbb{F}_{q_0}}$, since they are both in $\mathcal{C}$ and $\mathbb{F}_{q_0}^n$. Additionally $c \in \mathcal{C}|_{\mathbb{F}_{q_0}}$ implies that $kc \in \mathcal{C}|_{\mathbb{F}_{q_0}}$ for all $k \in \mathbb{F}_{q_0}$, since $kc \in \mathcal{C}$ and $kc \in \mathbb{F}_{q_0}^n$. Hence $\mathcal{C}|_{\mathbb{F}_{q_0}}$ is a $\mathbb{F}_{q_0}$ linear code.

Proposition 5.11. *If $\mathcal{C}$ is a $\mathbb{F}_q$ linear code and $\mathbb{F}_{q_0}$ is a subfield of $\mathbb{F}_q$, then $d(\mathcal{C}) \leq d(\mathcal{C}|_{\mathbb{F}_{q_0}})$.*

Proof. We have $d(\mathcal{C}) = \min_{c \in \mathcal{C}} \mathrm{wt}(c) \leq \min_{c \in \mathcal{C}|_{\mathbb{F}_{q_0}}} \mathrm{wt}(c) = d(\mathcal{C}|_{\mathbb{F}_{q_0}})$ since $\mathcal{C}|_{\mathbb{F}_{q_0}} \subseteq \mathcal{C}$. ■

If $\mathcal{C}$ is a $[n,k]_q$ code we generally do not have $d(\mathcal{C}) = d(\mathcal{C}|_{\mathbb{F}_{q_0}})$, where $\mathbb{F}_{q_0}$ is a subfield of $\mathbb{F}_q$. This is illustrated with the following example.

Example 5.12. We once again consider the finite field $\mathbb{F}_4$. Consider the $[3,2]_4$ linear code $\mathcal{C}$ with generator matrix:

$$G = \begin{bmatrix} \alpha & 0 & 1+\alpha \\ 1 & 1 & 1 \end{bmatrix}$$

We will show that $\mathcal{C}|_{\mathbb{F}_2}$ is nothing but the $[3,1]_2$ repetition code. Hence we assume that $c \in \mathcal{C}|_{\mathbb{F}_2}$, then

$$c = x_1 G_{*,1} + x_2 G_{*,2} \tag{5.5}$$

for some $x \in \mathbb{F}_4$. Notice that since $c_2 \in \mathbb{F}_2$ we must have $x_2 \in \mathbb{F}_2$, as $G_{0,1} = 0$ and $G_{0,2} = 1$. We will show that we must have $x_1 = 0$. Combining the $x_1 G_{*,1} = \begin{bmatrix} x_1\alpha & 0 & x_1 + \alpha x_1 \end{bmatrix}$ with Equation 5.5 we get that we must have:

$$x_1 + x_2 \in \mathbb{F}_2 \text{ and } (x_1 + \alpha x_1) + x_2 \in \mathbb{F}_2$$

The condition that $x_1 + x_2 \in \mathbb{F}_2$ yields that $x_1 \in \mathbb{F}_2$, since if $x_2 = 1$, then $1 + x_1 \in \mathbb{F}_2$ if and only if $x_1 \in \mathbb{F}_2$ as $1 + \alpha \notin \mathbb{F}_2$ and $1 + (1 + \alpha) = \alpha \notin \mathbb{F}_2$. Additionally the condition that $(x_1 + \alpha x_1) + x_2 \in \mathbb{F}_2$ implies that $x_1 = 0$, since $x_1 = 1$ gives $1 + \alpha + x_2 \in \mathbb{F}_2$ which would imply that $x_2 \in \{\alpha, 1+\alpha\}$. Hence $\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$ is a generator matrix for $\mathcal{C}|_{\mathbb{F}_2}$. Meaning $\mathcal{C}|_{\mathbb{F}_2}$ is nothing but the $[3,1]_2$ repetition code. Hence $d(\mathcal{C}|_{\mathbb{F}_2}) = 3$ however $d(\mathcal{C}) = \min\{\mathrm{wt}(c) | c \in \mathcal{C} \setminus \{0\}\} \leq 2$ since $\mathrm{wt}(G_{*,1}) = 2$. □

Remark 5.13. If we have a t-error correcting decoder for $\mathcal{C}$, then we may apply it on $\mathcal{C}|_{\mathbb{F}_{q_0}}$, however we might have $d(\mathcal{C}|_{\mathbb{F}_{q_0}}) > d(\mathcal{C})$, confer Example 5.12. Hence there might exist a decoding algorithm with a higher error correcting radius.

Theorem 5.14. *Let $x \in \mathbb{F}_q^n$ and $f \in \mathbb{F}_q[X]$ such that $f(x_i) \neq 0$ for all $i = 1, 2, \ldots, n$. Let $\mathbb{F}_{q_0}$ be a subfield of $\mathbb{F}_q$ then:*

(i) *The minimum distance of $\Gamma(x, f, \mathbb{F}_{q_0})$ atleast* $\deg(f) + 1$.

(ii) *Any word $y = c + e$ where $c \in \Gamma(x, f, \mathbb{F}_{q_0})$ and $e \in \mathbb{F}_q^n$ with* $\mathrm{wt}(e) \leq \left\lfloor \frac{\deg(f)+1}{2} \right\rfloor$ *can a decoded using either one of Algorithms 4.4 and 4.19.*

Proof. We start by showing Assertion (i). Note that since $\overline{\mathbb{F}}_q = \bigcup_{k=1}^{\infty} \mathbb{F}_{q^k}$ there exists an $N \in \mathbb{N}$ such that f splits into irreducible factors over $\mathbb{F}_{q^N}$, hence $d\left(\Gamma(x, f, \mathbb{F}_{q^N})\right) = \deg(f)+1$ by Corollary 5.8. The rest follows by Proposition 5.11.
Assertion (ii) follows since $\Gamma(x, f, \mathbb{F}_{q_0})$ is a subfield subcode of $\Gamma(x, f, \mathbb{F}_{q^N})$. Furthermore $\Gamma(x, f, \mathbb{F}_{q^N})$ has $\left\lfloor \frac{\deg(f)+1}{2} \right\rfloor$-error decoding algorithms by Theorems 4.6 and 4.21. ■

Appendices

A NP-Complete Problems

Our discussion of **NP**-complete problems will be based on Berlekamp et al. (1978). Since the topic of computational complexity theory is vast and not one of the primary topics of the project, the contents of this appendix will be quite informal. For a formal introduction we instead refer the reader to Sanjeev Arora (2009).

We start by introducing the notion of a *nondeterministic algorithm*. A nondeterministic algorithm is an algorithm which when presented with a choice between k alternative execution paths, are able to create k copies of it self, and simultaneously follow each of these k paths. The algorithm is said to solve the a problem if any one of these copies produces the right answer. We note that this repeated splitting may lead to an exponentially growing number of copies.

The problem space **NP** is defined to be the class of problems which are solvable by a nondeterministic algorithm whose time complexity is bounded by a polynomial in the length of the input (in the case of Problem 1.3, this will be n).

In contrast the problem space **P** is defined to be the class of all problems which can be solved by a deterministic algorithm whose time complexity is bounded by a polynomial in the length of the input.

Clearly $\mathbf{P} \subseteq \mathbf{NP}$, however it has not yet been proven whether $\mathbf{P} = \mathbf{NP}$ or $\mathbf{P} \neq \mathbf{NP}$, this problem was first formulated in 1971 and is one of the remaining millennium price problems, see Carlson et al. (2006). If $\mathbf{NP} = \mathbf{P}$, then there exists deterministic algorithms for solving any **NP** problem, in polynomial time. In particular the general decoding problem could be solved efficiently, rendering the McElice PKCS vulnerable to attack.

A problem is called **NP**-hard if it can be translated to a problem in **NP** in polynomial time.

Every problem in **NP** can be reduced to a problem called the *satisfiability problem*, meaning that if a deterministic polynomial time algorithm for the satisfiability problem is discovered, then it can be modified to solve every problem in **NP** (also in polynomial time). A problem for which the reverse is true, meaning that a deterministic polynomial time algorithm for the problem would yield a deterministic polynomial time algorithm for the satisfiability problem, is called **NP**-complete.

Hence if an algorithm for solving any one of these **NP**-complete problems was discovered, then it would yield polynomial time algorithms for solving any problem in **NP**. The **NP**-complete problems yields strong empirical evidence that $\mathbf{P} \neq \mathbf{NP}$, as no one has yet to find a algorithm with runs in polynomial time for a single of these well known problems. Before concluding our discussion on the class **NP**, we need to emphasize that the evidence for their hardness is purely empirical!

B Results from Previus Projects.

This appendix contains some results from previous projects which we will reference throughout the project. We will not provide proofs instead we refer the reader to Nørbjerg (2023) or Fulton (2008).

A proof of the following Proposition is provided in Nørbjerg (2023)[Proposition 2.88]

Proposition B.1. *Let $\mathcal{X}$ be a smooth projective algebraic curve and D be a divisor on $\mathcal{X}$, then:*

$$\ell(D) \leq 1 + \deg(D)$$

Theorem B.2 (Riemann-Roch Theorem)**.** *Let $\mathcal{X}$ be a smooth projective curve of genus g and D be a divisor on $\mathcal{X}$ then:*

$$\ell(D) - \ell(D - W) = \deg(D) - g + 1$$

for all canonical divisors W on $\mathcal{X}$.

Bibliography

E. Barker. Recommendation for key management, 2020. Avalible at https://doi.org/10.6028/NIST.SP.800-57pt1r5.

E. Berlekamp, R. McEliece, and H. van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Transactions on Information Theory*, 24(3):384–386, 1978. doi: 10.1109/TIT.1978.1055873.

J. Carlson, A. Jaffe, and A. Wiles. *The Millennium Prize Problems*. American Mathematical Society, 1st edition, 2006. ISBN 9781470476014.

A. Couvreur and H. Randriambololona. Algebraic geometry codes and some applications, 2020.

W. Fulton. Algebraic curves: An introduction to algebraic geometry, 2008. Avalible at https://dept.math.lsa.umich.edu/~wfulton/CurveBook.pdf.

W. C. Huffman and V. Pless. *Fundementals of Error-Correcting Codes*. Cambrige University Press, 1st edition, 2003. ISBN 9780521131704.

N. Koblitz. *A Course in Number Theory and Cryptography*. Springer-Verlag, 2nd edition, 1994. ISBN 0387942939.

N. Lauritzen. *Concrete Abstract Algebra: From Numbers to Gröbner Bases*. Cambridge University Press, 1st edition, 2003. ISBN 9780521534109.

P. J. Lee and E. F. Brickell. An observation on the security of mceliece's public-key cryptosystem. In *Advances in Cryptology — EUROCRYPT '88*, pages 275–280, Berlin, Heidelberg, 1988. Springer Berlin Heidelberg. ISBN 978-3-540-45961-3.

C. Löndahl. *Some Notes on Code-Based Cryptography*. PhD thesis, Lund University, 2014.

M. S. Nørbjerg. Algebraic geometry codes. Bachellor's thesis, Aalborg Univeristy, June 2023.

I. Panaccione. *On decoding algorithms for algebraic geometry codes beyond half the minimum distance*. PhD thesis, Institut Polytechnique de Paris, 2021.

R. Pellikaan, X.-W. Wu, S. Bulygin, and R. Jurrius. *Codes, Cryptology and Curves with Computer Algebra*. Cambrige University Press, 1st edition, 2018. ISBN 9780521520362.

C. Peters. Information-set decoding for linear codes over $\mathbb{F}_q$. In N. Sendrier, editor, *Post-Quantum Cryptography*, pages 81–94, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-12929-2.

B. B. Sanjeev Arora. *Computational Complexity A modern Approach*. Cambridge University Press, 1st edition, 2009. ISBN 9780521424264.

P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, oct 1996.

H. Singh. Code based cryptography: Classic mceliece, 2020.

H. Stichtenoth. *Algebraic Function Fields and Codes*. Springer Berlin, Heidelberg, 2nd edition, 2009. ISBN 9783540768784.