

MASTER THESIS

A Finite Geometry Construction for MDPC-Codes

A thesis submitted in partial fulfilment of the
requirements for the degree of Master of Science in
Mathematics

Author:

Jessica Bariffi

Supervised by:

Prof. Dr. Joachim Rosenthal

Co-supervised by:

Dr. Alessandro Neri



**Universität
Zürich** ^{UZH}

Department of Mathematics

University of Zurich

April 28, 2020

Abstract

This master thesis presents a new construction of MDPC-codes. MDPC-codes are binary linear codes of length n characterized by a parity check matrix of row-weight $\mathcal{O}(\sqrt{n})$. We provide a construction of a parity check matrix for these MDPC-codes using finite geometry. To be precise, we work in the Desarguesian projective plane $PG(2, q)$ of order q , where q is an odd prime power. The projective plane is an incidence structure consisting of a set of points and a set of lines united with a set of non-degenerate conics. These non-degenerate conics are mutually intersecting in a unique point of the Desarguesian plane and we call the collection of all these non-degenerate conics a projective bundle. We give a full characterization of the, in total, three types of projective bundles in this master thesis and discuss its existence and how we are able to use them. In fact the non-degenerate conics of a projective bundle in $PG(2, q)$ can be interpreted as lines which yields the structure of a projective plane. To simplify the idea we can say that we work with the distinct union of two projective planes, namely Π consisting of a set of points and a set of lines and Γ consisting of a set of points and a projective bundle in the Desarguesian plane. The parity-check matrix that we present is the concatenation of the two incidence matrices H' and H'' of the two projective planes Π and Γ , respectively. It characterizes the family of MDPC-codes $C_2(\Pi \sqcup \Gamma)^\perp$ of length $2(q^2 + q + 1)$ and dimension $q^2 + q + 2$. Due to the fact that lines intersect a non-degenerate conic at most in two points, two randomly chosen columns of the parity check matrix share at most two positions where they both have a non-zero entry. The smaller this number of positions is, the better the error-correction performance becomes. Indeed, applying one round of Gallager's bit-flipping algorithm to a parity check matrix constructed in that way implies that we are able to correct $\lfloor \frac{q+1}{4} \rfloor$ errors in a received word. Additionally we have lower bounded the minimum distance of these MDPC-codes $C_2(\Pi \sqcup \Gamma)^\perp$ by $\left\lceil \frac{2(q+2)}{3} \right\rceil$.

Acknowledgments

I would like to thank my supervisor Prof. Dr. Joachim Rosenthal for giving me the opportunity to write the master thesis in coding theory and, in addition, for his steady support to reach my future goal. I feel grateful for everything I learned about coding theory and cryptography through his courses held at University of Zurich, which laid the foundation of my fascination for this area.

Secondly I would like to express my sincerest gratitude to Dr. Alessandro Neri, who guided me through the world of error-correcting codes. I thank him for his inputs and ideas and for offering me a helping hand whenever needed, even when he stayed in another country. I feel extremely happy to have had a co-supervisor like him.

On a personal level I would love to express thanks to all my great fellow students that I consider friends by now. I am filled with gratitude that I had an incredible group of friends studying with me. Last but not least I would like to thank my family and Raphael Imfeld for the never ending belief in my forte that helped me improve from day to day.

Contents

1	Introduction	2
2	Background in Coding Theory	4
2.1	Goals of Coding Theory	4
2.2	Block Codes and Their Parameters	5
2.3	Linear Block Codes	8
2.3.1	Finite Fields	8
2.3.2	Definitions and Representations	9
2.3.3	Distance and Weight	11
2.4	Encoding and Decoding	13
3	Introduction to Finite Geometry	16
3.1	Incidence Structures	16
3.2	Finite Projective Spaces	18
3.3	Projective Planes	20
3.4	Arcs and Ovals	23
3.5	Codes from Projective Planes	26
4	Projective Bundles	28
4.1	Definitions and Basic Results	28
4.2	Existence of Projective Bundles	30
4.3	Geometrical Representation in $PG(5, q)$	32
4.4	Classification	34
4.5	Alternative Representation	41
5	Moderate Density Parity-Check Codes	46
5.1	Motivation and Definitions	46
5.2	Decoding Algorithms	47
5.2.1	The Bit-Flipping Algorithm	48
5.2.2	The Gallager A Algorithm	51
5.2.3	Belief Propagation	52
5.3	Error-Correction Capacity	52
5.4	McEliece Cryptosystem	54
6	Construction	56
6.1	Parity-Check Matrix	57
6.2	Dimension	60
6.3	Minimum Distance	61
6.4	Error-Correction Capability	66
7	Conclusion	71
A	Sage Functions	73

Chapter 1

Introduction

There exist many public key cryptosystems based on factorization of large numbers or based on the so-called discrete logarithm. One of the most famous and established examples of such a cryptosystem is RSA. Nowadays all these public key cryptosystems can be attacked in polynomial time by a quantum computer (for details see [32] and [6]). Of course quantum computers do not exist for now but they are expected to exist in the near future. Mathematicians and cryptographers are trying to develop new cryptosystems that are resistant against quantum computers. Code-based cryptography is believed to be quantum-resistant and is hence considered a judicious solution in future applications. Code-based cryptosystems offer a lot more advantages other than being quantum-resistant. They also show a high algorithmic efficiency. The first code-based cryptosystem is the McEliece cryptosystem published in 1978 [23]. McEliece proposed this system using binary Goppa codes. Even if it provided fast encryption and decryption, one disadvantage of the cryptosystem is its extremely large key size. So the goal was then to reduce the key size. This was achieved with the introduction of Low-density parity-check codes (LDPC-codes) in 1963 by Robert Gallager [10]. These codes are binary linear error-correcting codes with a parity-check matrix that consists only of a few nonzero entries per column and row. Despite the efficient error-correction performance, LDPC-codes can not be used in the way they were proposed for the McEliece cryptosystem. Since the parity-check matrix of an LDPC-code is the generator matrix of its dual code, the codewords of its dual code are of low weight and can easily be computed and used to identify the rows of the parity-check matrix.

Therefore the idea of LDPC-codes needed to be modified. This is where Moderate-density parity-check codes (MDPC-codes) were introduced. The parity-check matrix of these codes is not sparse anymore, but its row weight is of order $\mathcal{O}(\sqrt{n})$, where n is the length of the code, which seems to prevent the attacks used on LDPC-codes. MDPC-codes were used instead of Goppa codes in the McEliece cryptosystem and there have been various constructions based on quasi-cyclic structures or randomness (for example in [24], [37]). However, the error-correction performance of MDPC-codes has slightly suffered when compared to LDPC-codes. There have been several improvements of the error-correction performance using a modified version of Gallagers bit-flipping algorithm [37], [18]. Depending on the construction of the code and the decoding algorithm used the performance varies.

The aim of this master thesis is to develop a construction based on finite geometry to

improve the error-correction capability when performing one round of bit-flipping. We make use of the idea of constructing codes from projective planes, characterized by their incidence matrices. We consider the natural Desarguesian plane $PG(2, q)$ of odd order q which consists of a set of points and a set of lines. Moreover we study packings of non-degenerate conics in $PG(2, q)$, which are sets of $q + 1$ points satisfying a quadratic equation. Such a packing of conics in the Desarguesian plane, also called projective bundles, together with the set of points is again a projective plane. Hence a conic in $PG(2, q)$ can be interpreted as a line. Furthermore a line intersects a conic in at most two points. We make use of this property in order to provide a good error-correction capacity (see section 6.4).

Chapter 2 and 3 give a short introduction to coding theory and to finite geometry, respectively. There we define and state the most important concepts and results that will be useful in the course of the thesis.

In Chapter 4 we discuss the structure of projective bundles and we will see under which circumstances they do, or do not, exist. We will also give some important properties and behaviours of projective bundles which will be of great use in Chapter 6.

An introduction of MDPC-codes will then be given in Chapter 5. We will explore some decoding algorithms that can be used for MDPC-codes with the main focus on the bit-flipping algorithm. For this specific algorithm we will also discuss the performance when used with MDPC-codes instead of LDPC-codes. We will quickly discuss one variant of the McEliece cryptosystem using MDPC-codes to get an idea of how MDPC-codes can be used in practice.

Finally, in Chapter 6 we present a new construction of MDPC-codes based on finite geometry. More explicitly, we construct a parity-check matrix that characterizes an MDPC-code. For this family of codes we determine its dimension and we provide upper and lower bounds on the minimum distance. Furthermore, we discuss the error-correction capability of the code using the bit-flipping algorithm.

Chapter 2

Background in Coding Theory

We start with a preliminary Section on coding theory to introduce the most basic definitions and notions needed in the course of the following pages. The theory has been taken mainly from Guruswami, Rudra and Sudan [14, Part I, Chapter 1, 2.1 and 2.2, pp. 17-44], from Assmus and J.D. Key [1, Chapters 1 and 2] and from Lindell [20, Chapters 1 and 2].

2.1 Goals of Coding Theory

In our lives communication plays a central role. We are trying to send messages to our person opposite to tell him or her what we want. Sometimes, even though we do not find explicit words or if we are surrounded by too much noise, the other person is able to understand what we are saying. This shows that a language has some redundancy that allows to communicate even if some small errors occur. Not only in communication between humans, but also in the digital world redundancy is used to correct errors.

One goal of coding theory is to improve the trustworthiness of a communication. To achieve this, coding theory deals with error-correcting codes to add redundancy. In 1948 Claude Elwood Shannon published a paper (see [31]), in which he showed that for a given degree of noise in a communication channel one can communicate nearly error-free. Error-correcting codes were then introduced to implement this theorem. The message or data sent is first encoded, which means that it will be transformed into a codeword, by adding redundancy. After the message is sent through the channel, the receiver decodes the received word into the original message. Of course there is the possibility of errors occurring during the transmission, meaning that possibly not every symbol of the message is transmitted correctly. We call these possibly changed symbols *errors*. The goal is to add as little redundancy as possible needed to correct a certain amount of errors in a code. So the central question that arises is: How much redundancy is needed?

At the same time, we are trying to efficiently encode some information by not using much space. Indeed, one can achieve this by removing redundancy from the received information. Thus, the idea is to find a clever way of introducing redundancy in order to encode efficiently but at the same time one has to be able to correct as many errors as possible.

Finally, coding theory is also used in cryptography since we want to construct systems that are secure but that are also usable in noisy channels.

2.2 Block Codes and Their Parameters

First and foremost we introduce the general notions of coding theory such as the definition of a code and its parameters. In Section 2.3 we then focus on a specific class of codes, which are of interest in the course of this thesis.

Let $A = \{a_1, \dots, a_q\}$ denote a finite set of q elements which we call an *alphabet of q letters*. Furthermore, let M be a finite set of information words and define a map

$$\varphi : M \longrightarrow A^n.$$

If φ is an injective map we call $C := \text{Im}(\varphi)$ a *q -ary block code of block length n* with *encoder φ* . So a *q -ary block code C* is a subset of A^n . The elements of a code C , called *codewords*, are n -tuples $c = (c_1, \dots, c_n) \in A^n$. The ambient space A^n can be viewed as a set of vectors, if the set A admits some structure, as we will see for linear codes in the next Section 2.3. In other situations the ambient space can also be viewed as a set of sequences or as a set of functions.

Another parameter which is important for the description of a code is its dimension. Given a q -ary block code $C \subset A^n$ we define the *dimension k* of C by

$$k = \log_q |C|.$$

As already mentioned, coding theorists are interested in the amount of redundancy needed. Redundancy, informally speaking, is the amount of superfluous space needed to transmit a certain message or data through a channel. It is the part of the message that contains no information but which is used to detect errors in the transmission. Therefore, we want to define the *redundancy* of a block code of length n and dimension k as the difference $n - k$. The importance of the amount for redundancy motivates the definition of a measure of redundancy. We define the *rate*, denoted $R(C)$, of an $[n, k]$ -block code C to be the ratio

$$R(C) = \frac{k}{n}.$$

We can say that the rate of a code measures the amount of useful information in all of the n symbols. Note that $k \leq n$ and hence $R(C) \leq 1$.

As mentioned, we would like to have an optimal error-correction performance. Usually a codeword is expected to contain less errors than expected. To be able to discuss about the number of errors in a codeword we want to give the definition of the Hamming distance.

Definition 2.2.1. [14, Definition 1.4.1] Consider a finite field \mathbb{F} . Let $x, y \in \mathbb{F}^n$. Then the *Hamming distance* $d(x, y)$ is the number of positions in which x and y differ, i.e.

$$d(x, y) = |\{i \in \{1, \dots, n\} \mid x_i \neq y_i\}|.$$

In fact the Hamming distance is dependent on the positions i in which x and y differ and not in the actual values of x_i and y_i . Therefore the following result holds for a finite field \mathbb{F} (note that finite fields will be introduced in the next Section).

Lemma 2.2.2. [20, Proposition 1.5] *The Hamming distance $d : \mathbb{F}^n \times \mathbb{F}^n \rightarrow \mathbb{N}$ defines a metric.*

Proof. First, we show that the Hamming distance is positive definite. Let $x, y \in C$ be arbitrary codewords. The Hamming distance of x and y is the number of entries in which x and y differ

$$d(x, y) = |\{i \in \{1, \dots, n\} \mid x_i \neq y_i\}|.$$

Therefore, it has to be non-negative. Further, the Hamming distance is zero if and only if there is no position in which x and y differ. Then x and y must be the same codeword.

Secondly, we need to prove that the Hamming distance is symmetric. Again let x and y be arbitrary codewords of C . The definition gives that

$$d(x, y) = |\{i \in \{1, \dots, n\} \mid x_i \neq y_i\}|,$$

which is equivalent to

$$|\{i \in \{1, \dots, n\} \mid y_i \neq x_i\}| = d(y, x).$$

Therefore, we obtain that the Hamming distance is symmetric.

Lastly, we need to show, that the triangle inequality¹ holds for any codewords x, y and z of the code C . We do this by induction on the code length n . For this we define S_n to be the set of sequences of bits of length n . For all $x, y \in S_n$ let $d_n(x, y)$ be the number of positions in which x and y differ.

For $n = 1$ we have $S_1 = \{0, 1\}$ and therefore d_1 is the discrete metric for S_1 . Hence, it satisfies the triangle inequality.

Now assume that for arbitrary but fixed $n - 1$, the distance d_{n-1} satisfies the triangle inequality. We must then deduce that also d_n does.

Let $x, y, z \in S_n$. Decompose x into its first bit x' and its last $n - 1$ bits x'' , so that $x = x'x''$. Similarly we can decompose $y = y'y''$ and $z = z'z''$. Then the distance $d_n(x, z)$ is equal to $d_1(x', z')$ plus the number of positions in which x'' and z'' differ, which is

$$d_n(x, z) = d_1(x', z') + d_{n-1}(x'', z'').$$

Since d_1 and d_{n-1} both satisfy the triangle inequality, we obtain by the induction hypothesis

$$\begin{aligned} d_n(x, z) &\leq d_1(x', y') + d_1(y', z') + d_{n-1}(x'', y'') + d_{n-1}(y'', z'') \\ &= d_1(x', y') + d_{n-1}(x'', y'') + d_1(y', z') + d_{n-1}(y'', z'') \\ &= d_n(x, y) + d_n(y, z). \end{aligned}$$

Hence we conclude that the Hamming distance satisfies the triangle inequality. \square

¹The triangle inequality for the Hamming distance $d : \mathbb{F}_2 \times \mathbb{F}_2 \rightarrow \mathbb{N}$ is satisfied, if for any $x, y, z \in C$ it holds $d(x, z) \leq d(x, y) + d(y, z)$

Another important quantity is the *minimum distance* of a code C , denoted $d(C)$, which is defined to be the smallest possible Hamming distance between two codewords,

$$d(C) := \min\{d(c, \tilde{c}) \mid c, \tilde{c} \in C, c \neq \tilde{c}\}.$$

Knowing the minimum distance of a code C , one can estimate the number of errors that can be detected as well as the number of errors that can be corrected. Indeed, the higher the minimum distance of a code, the more errors can be detected and corrected. The following result summarizes these facts.

Proposition 2.2.3. [14, Proposition 1.4.1] *Let $C \subset GF(q)^n$ be a code of length n with minimum distance $d(C) \geq 2$. Then the following hold:*

- i) If $d(C) = 2t + 1$ for some $t \in \mathbb{N}$, then C can correct up to t errors,*
- ii) C can detect up to $d(C) - 1$ errors,*
- iii) C can correct up to $d(C) - 1$ erasures.*

Proof. Assume first that C has an odd minimum distance $d(C) = 2t + 1$. Suppose that a codeword $c_1 \in C$ is transmitted and $y \in GF(q)^n$ is received with $c_1 \neq y$. Since t is the number of errors we note that

$$d(c_1, y) \leq t. \tag{2.1}$$

We claim now, that $c_1 \in C$ is the only codeword satisfying (2.1). Indeed, if $c_2 \in C$ is another codeword with $d(c_2, y) \leq t$ then by the triangle inequality we obtain

$$\begin{aligned} d(c_1, c_2) &\leq d(c_1, y) + d(y, c_2) \\ &\leq 2t \\ &= d(C) - 1. \end{aligned}$$

Since we have assumed that $d(C) = 2t + 1$ it follows that $c_1 = c_2$.

To prove the second statement assume again that a codeword $c \in C$ is sent and y is received such that $y \neq c$. We need to check, if the received word is a codeword of C . If it is a codeword of C , then the errors could have been detected. On the other hand if $d(c, y) \leq d(C) - 1$, then y is not a codeword of C .

For the last statement let $y \in (\mathbb{F} \cup \{?\})^n$ be a received codeword, where $?$ denotes the erasure. We claim that there is a unique codeword $c \in C$ that agrees with y . Indeed, if there are two codewords $c_1, c_2 \in C$ agreeing with y in the unerased positions, then

$$d(c_1, c_2) \leq |\{i = 1, \dots, n \mid y_i = ?\}| \leq d(C) - 1,$$

which is impossible since $d(C)$ is the minimum possible distance between two codewords of C . Hence, y is still a unique codeword of C . \square

2.3 Linear Block Codes

We will now introduce a family of codes called linear codes. They have more structure than other codes and therefore admit some nice properties. We will explore how this family of codes can be represented using matrices and how these matrices are used to determine the minimum distance of a linear code. Later in this thesis we will exclusively focus on binary linear codes.

2.3.1 Finite Fields

In order to define a linear code we need the definition of a finite field. Finite fields were discovered by the mathematician Évariste Galois. In his honour, finite fields are often called Galois fields and are denoted by $GF(q)$, where q is the number of elements in this field. More commonly, in the literature, the notation \mathbb{F}_q is also used for a finite field of q elements. These fields play an important role in cryptography and coding theory, since we deal with a finite amount of symbols, signs or numbers.

Recall that a field \mathbb{F} is a triple $(S, +, \cdot)$ consisting of a set of elements S and two functions $+/\cdot : \mathbb{F} \times \mathbb{F} \rightarrow \mathbb{F}$, both satisfying the following properties:

- (F1) The functions are both well-defined, i.e. for any two elements $x, y \in S$ it holds that $x + y \in S$ and $x \cdot y \in S$.
- (F2) Both functions $+$ and \cdot are associative, i.e. for any $x, y, z \in S$ it holds that $(x + y) + z = x + (y + z)$ and $(x \cdot y) \cdot z = x \cdot (y \cdot z)$.
- (F3) Both functions $+$ and \cdot are commutative, i.e. for any $x, y \in S$ it holds that $x + y = y + x$ and $x \cdot y = y \cdot x$.
- (F4) The functions follow the distributivity law, i.e. for any $x, y, z \in S$ it holds that $(x + y) \cdot z = x \cdot z + y \cdot z$.
- (F5) For each of the functions there exists a neutral element, 0 or 1 respectively, such that for every $x \in S$ we have $x + 0 = x$ and $x \cdot 1 = x$.
- (F6) For every element $x \in S$ there is a unique additive inverse $(-x)$ such that $x + (-x) = 0$ and for each nonzero $y \in S$ there is a multiplicative inverse y^{-1} such that $y \cdot y^{-1} = 1$.

A *finite field* is a field consisting of a finite number of elements. We will denote such a field by $GF(q)$, where q is the number of elements in that field. In fact, the number of elements is always a prime power as the following theorem states.

Theorem 2.3.1. [14, Theorem 2.1.1] *The size of any finite field $GF(q)$ is $q = p^k$, where p is a prime number and $k \geq 1$ an integer.*

Finite fields do not exist for any choice of q . In fact, they only exist for prime powers. The following result clarifies this question.

Theorem 2.3.2. [14, Theorem 2.1.3] *For every prime power q there is a unique finite field with q elements (up to isomorphism²).*

Example 2.3.3. A well-known example for finite fields are the fields formed by the integers modulo p for a prime number p . We denote this field by $GF(p) = (\{0, 1, \dots, p-1\}, +_p, \cdot_p)$, where $+_p$ and \cdot_p denote the addition and multiplication modulo p , respectively.

It is well-known that the multiplicative group (F^*, \cdot_q) of the field $GF(q) = (F, +_q, \cdot_q)$ is isomorphic to the cyclic group of order $q-1$, i.e.

$$(F^*, \cdot_q) \cong (Z/\langle q-1 \rangle, +).$$

This leads to the definition of a generator or primitive element of $GF(q)$.

Definition 2.3.4. [11, Definition 0.2.2] A *generator* of the multiplicative group of $GF(q)$ is an element of order $q-1$.

We can define a *linear subspace* $S \subset GF(q)^n$ to be a non-empty set satisfying the following two properties:

- i) For every element $x, y \in S$, $x + y \in S$.
- ii) For every $x \in S$ and every scalar $a \in GF(q)$, $a \cdot x \in S$.

Example 2.3.5. Consider the finite field $GF(3)^2$, then the set $S = \{(0, 0), (1, 1), (2, 2)\}$ is a simple example for a linear subspace of $GF(3)^2$.

Finally, a *semi-field* is a non-associative division-ring. Hence, a semi-field differs from a field in that their multiplication is not required to be commutative.

2.3.2 Definitions and Representations

Now having the theory of finite fields ready, we can define a linear code. For this, let $GF(q)$ always denote a finite field of q elements, where $q = p^k$ is a power of the prime p .

Definition 2.3.6. [14, Definition 2.0.1] A *q -ary linear code* C of length n and dimension k is a linear subspace of $GF(q)^n$ of dimension k .

Therefore, the length of a q -ary linear code C can be derived directly from the dimension of the ambient space $GF(q)^n$. Recall further from Section 2.2 that the dimension of a code C is $k = \log_q |C|$. The *dimension* of C , denoted by $\dim(C)$, is the dimension of C as a vector space of $GF(q)^n$.

We denote a q -ary linear code C by $[n, k]_q$ -linear code.

Since we know that the dimension $k = \dim(C)$ is $\log_q |C|$, the number of elements in an $[n, k]_q$ -linear code is given by

$$|C| = q^k.$$

²An *isomorphism* is a bijective map $\varphi : A \rightarrow B$ between two mathematical structures $(A, +_A, \cdot_A)$ and $(B, +_B, \cdot_B)$ such that for every $x, y \in A$ it holds $\varphi(x +_A y) = \varphi(x) +_B \varphi(y)$ and $\varphi(x \cdot_A y) = \varphi(x) \cdot_B \varphi(y)$.

For example, a binary linear code of dimension k has 2^k codewords.

We can represent an $[n, k]_q$ -linear code C using matrices. In order to do so, we need to define a dual code as well.

Definition 2.3.7. [20, Definition 2.4] Let C be an $[n, k]_q$ -linear code. Its *dual code* C^\perp is given by

$$C^\perp = \{x \in GF(q)^n \mid x \cdot c^\top = 0, \forall c \in C\}.$$

Since C is a subspace of $GF(q)^n$, it can be represented by a basis. Therefore, we define a *generator matrix* G of the code to be a $(k \times n)$ -matrix over $GF(q)$ whose rows are formed by any k linearly independent vectors of C , that form a basis of C . Similarly, we define a matrix $H \in GF(q)^{(n-k) \times n}$, the *parity-check matrix* of C , to be the generator matrix of the dual code C^\perp .

The following result gives a necessary and sufficient condition for a matrix H to be a parity-check matrix of an $[n, k]_q$ -linear code C .

Lemma 2.3.8. [20, Lemma 2.14] *Let C be an $[n, k]_q$ -linear code with generator matrix G . Then a vector $v \in GF(q)^n$ is a codeword of the dual code C^\perp if and only if $v \cdot G^\top = 0$. In particular, an $((n-k) \times n)$ -matrix H over $GF(q)$ is a parity-check matrix of the code C if and only if its rows are linearly independent and it satisfies $H \cdot G^\top = 0$.*

Proof. Let us start by proving the first part. For this, assume that an arbitrary vector $v \in GF(q)^n$ is a codeword of the dual code C^\perp . Denote the rows of the generator matrix G by g_1, \dots, g_k , where each $g_i \in GF(q)^n$. Since the rows of G form a basis for the code C , for each codeword $c \in C$ there exist k scalars $\lambda_1, \dots, \lambda_k \in GF(q)$ satisfying

$$c = \sum_{i=1}^k \lambda_i \cdot g_i.$$

Since $v \in C^\perp$, and by the definition of the dual code, it holds that $v \cdot g_i^\top = 0$ for each $i = 1, \dots, k$. Hence, $v \cdot G^\top = 0$.

Conversely, assume that for an arbitrary vector $v \in GF(q)^n$ we have $v \cdot G^\top = 0$. This means that for every $i = 1, \dots, k$ $v \cdot g_i = 0$. Furthermore, let $c \in C$ be any codeword such that it holds that $c = \sum_{i=1}^k \lambda_i \cdot g_i$ for some $\lambda_1, \dots, \lambda_k \in GF(q)$. Then, we obtain

$$v \cdot c = \sum_{i=1}^k v_i \cdot \lambda_i \cdot g_i^\top = \sum_{i=1}^k \lambda_i \cdot (v_i \cdot g_i^\top) = \sum_{i=1}^k \lambda_i \cdot 0 = 0.$$

Thus we conclude $v \in C^\perp$.

For the second part of the statement let us assume that $H \in GF(q)^{(n-k) \times n}$ is a parity-check matrix of C . Hence, H is a generator matrix of C^\perp , which implies that its rows are linearly independent in C^\perp . Hence, by the above computations we have

$$H \cdot G^T = 0.$$

If conversely we assume that $H \cdot G^T = 0$, where H is any matrix in $GF(q)^{(n-k) \times n}$, then for each row v of H , and hence in C^\perp , we have that $v \cdot G^T = 0$. Since the rows are all linearly independent, H is a generator matrix of C^\perp and hence it is a parity-check matrix of C , which concludes the proof. \square

Note that this means that every matrix generating the dual C^\perp is a characterization of the code C . Equivalently to the statement above we could say, that a vector $c \in GF(q)^n$ is a codeword of an $[n, k]_q$ -linear code C with parity-check matrix H if and only if it satisfies $H \cdot c^T = 0$.

Example 2.3.9. Let C be a $[4, 2]$ binary linear code with parity-check matrix H given by

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

By the definition of a linear code, this code C has dimension $\dim(C) = 2$ and hence, by the above discussion, $2^2 = 4$ elements. Furthermore, we know that each element $c \in C$ has to satisfy $H \cdot c^T = 0$, which gives

$$\begin{pmatrix} c_3 + c_4 \\ c_1 + c_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Thus, every codeword $c = (c_1, c_2, c_3, c_4)$ of C has to satisfy $c_3 + c_4 = 0$ and $c_1 + c_2 = 0$. So we conclude that the codewords of C are the following four:

$$c_1 = (0, 0, 0, 0), \quad c_2 = (0, 0, 1, 1), \quad c_3 = (1, 1, 0, 0), \quad c_4 = (1, 1, 1, 1).$$

2.3.3 Distance and Weight

We have mentioned the general definition of the distance of a code C earlier in this preliminary Section on coding theory. For an $[n, k]_q$ -linear code there is another description, called the weight of a codeword. We will see that for linear codes the minimum distance and minimum weight are equivalent. In this subsection let C always denote an $[n, k]_q$ -linear code if nothing else is stated.

We define the *weight* of a vector $x = (x_1, \dots, x_n) \in GF(q)^n$ to be the number of non-zero positions of x , i.e.

$$wt(x) = |\{i = 1, \dots, n \mid x_i \neq 0\}|.$$

If every row x of a matrix H has a constant weight $wt(x) = w$ then we say that H has *row-weight* w . Similarly, we say that H has *column-weight* v , if each column of H has a constant weight v .

Lemma 2.3.10. [14, Proposition 2.3.4] *Let C be an $[n, k]_q$ -linear code of minimum distance $d(C)$. Then the distance is the minimum possible weight of the non-zero codewords, i.e.*

$$d(C) = \min\{wt(c) \mid c \in C, c \neq 0\}.$$

Proof. Consider two distinct non-zero codewords $x, y \in C$ that are a distance $d(x, y) = d$ apart. Since C is linear and binary, $x - y$ is also a codeword of C and has d non-zero elements. Therefore

$$d(x, y) = d(x - y, 0) = wt(x - y).$$

Thus we can say that the minimum distance $d(C)$ is equal to the minimum weight of all the $q^k - 1$ non-zero codewords. \square

In general, knowing the minimum weight of a code C does not give an exact result on the minimum weight of its dual code C^\perp . The following shows that the minimum distance of a code C can be determined using a parity-check matrix.

Theorem 2.3.11. [14, Proposition 2.3.5] *Let C be an $[n, k]_q$ -linear code with parity-check matrix H . Then C has minimum distance $d(C) \geq d$ if and only if every set of $d - 1$ columns of H is linearly independent.*

Proof. Assume first that the minimum distance is $d(C) \geq d$. Further assume that there is a set of $s < d$ columns of H , say h_1, \dots, h_s , that are linearly dependent. This means that there are constants $\lambda_1, \dots, \lambda_s$ not all zero such that

$$\sum_{i=1}^s h_i \cdot \lambda_i = 0.$$

The idea is to construct a codeword c of weight $wt(c) < d$. For this let $c = (\lambda_1, \dots, \lambda_s, 0, \dots, 0)$. We see that c is indeed a codeword of C because

$$H \cdot c^\top = \sum_{i=1}^s h_i \cdot \lambda_i = 0.$$

But $wt(c) = s < d$ which is a contradiction since $d(C) \geq d$.

Conversely, assume that every set of $d - 1$ columns of H is linearly independent and let $c = (c_1, \dots, c_n) \in C$ be a codeword of weight $wt(c) \leq d - 1$. Since c is a codeword of C it holds that

$$H \cdot c^\top = \sum_{i=1}^n h_i \cdot c_i = 0,$$

where h_1, \dots, h_n denote the columns of H . Hence, because $wt(c) < d - 1$, there exist $d - 1$ columns of H summing to zero. Thus there are $d - 1$ columns of H which are linearly dependent, which is a contradiction to the assumption. So we obtain that $wt(c) \geq d$ and since C is linear and $c \in C$ was chosen arbitrarily we conclude that $d(C) \geq d$. \square

We know now that the minimum distance gives the smallest difference between two codewords. Since it describes a minimum value, we are interested in the maximal possible distance among all $[n, k]_q$ -linear codes. In 1964 Richard Collom Singleton [35] has found an upper bound for the minimum distance of an arbitrary code C of length n and dimension k .

Theorem 2.3.12 (Singleton-bound, [35]). *For any $[n, k]_q$ -linear code C its minimum distance satisfies $d(C) \leq n - k + 1$.*

Proof. Assume that $C = \ker H$, where $H \in GF(q)^{(n-k) \times n}$ is a parity check matrix of C . We will make use of the previous theorem by giving the number of linearly dependent columns of H . Since H has $n - k$ rows and because the number of linearly dependent columns is equal to the number of linearly dependent rows, we obtain that at most any $n - k$ columns of H are linearly independent. Hence, we conclude by the previous Theorem that $d(C) \leq n - k + 1$. \square

A code C attaining this bound is called a *maximum distance separable code*, or shortly MDS code. In this thesis we will not focus on this family of codes, therefore we will not go into details here.

The following bound, which for binary linear block codes is even more precise, is the Griesmer bound which was published by Griesmer in 1960 in his article "A bound for Error-Correcting Codes" [12].

Theorem 2.3.13 (Griesmer-bound, [12]). *Let C be an $[n, k]_2$ -linear code. Then the following estimation holds*

$$n \geq \sum_{i=0}^{k-1} \left\lceil \frac{d}{2^i} \right\rceil.$$

2.4 Encoding and Decoding

As already mentioned in Section 2.1, coding theory is used to improve the communication. We said that a message is first encoded, which is the process of adding redundancy to transform a message into a codeword, and then decoded after the transmission. In this section we shortly describe encoding and decoding using the tools examined earlier.

The codewords of an $[n, k]_q$ -linear code are usually described as row vectors. An *encoding map* is defined by a $(k \times n)$ -generator matrix G of C as a linear map

$$\begin{aligned} \varphi : GF(q)^k &\longrightarrow GF(q)^n \\ m = (m_1, \dots, m_k) &\longmapsto m \cdot G = c = (c_1, \dots, c_n) \end{aligned}$$

mapping any message m to a codeword $c \in C$ and satisfying

$$C = \text{Im}(\varphi).$$

We define a *syndrome former* of C using a parity-check matrix H . It is defined by the linear map

$$\begin{aligned} \psi : GF(q)^n &\longrightarrow GF(q)^{n-k} \\ c = (c_1, \dots, c_n) &\longmapsto H \cdot c^\top, \end{aligned}$$

such that it holds that

$$C = \ker(\psi).$$

The syndrome former checks whether a vector $y \in GF(q)^n$ is a codeword or not.

Remark 2.4.1 ([5], Remark 1.9). An encoding map $\varphi : GF(q)^k \rightarrow GF(q)^n$ is necessarily injective and a syndrome former $\psi : GF(q)^n \rightarrow GF(q)^{n-k}$ is necessarily surjective.

Proof. Let us first prove, that φ is injective. By the rank nullity Theorem³ we know that

$$\dim(GF(q)^k) = \dim(\text{Im}(\varphi)) + \dim(\ker(\varphi)). \quad (2.2)$$

We have that $\dim(GF(q)^k) = k$ and by definition of the encoding map $\text{Im}(\varphi) = C$. Therefore we obtain

$$\dim(\text{Im}(\varphi)) = \dim(C) = k.$$

Using this and (2.2) we can conclude that $\dim(\ker(\varphi)) = 0$, hence φ is injective. Similarly for the syndrome former ψ we have

$$\dim(GF(q)^n) = \dim(\text{Im}(\psi)) + \dim(\ker(\psi)). \quad (2.3)$$

Again using the definition of a syndrome former of C , we get that

$$\dim(\ker(\psi)) = \dim(C) = k.$$

Substituting this in (2.3) we obtain

$$n = \dim(\text{Im}(\psi)) + k$$

and hence

$$\dim(\text{Im}(\psi)) = n - k = \dim(GF(q)^{n-k}).$$

Therefore we conclude that ψ is surjective. □

With the help of these two maps we can decode a linear code. For this, assume that $C \subset GF(q)^n$ is an $[n, k]$ -linear code of distance $d(C) = 2t + 1$. From proposition 2.2.3 it follows that we can correct up to t errors and detect up to $2t$ errors. Suppose that $\psi : GF(q)^n \rightarrow GF(q)^{n-k}$ is a syndrome former of C . Let us then look at the error correction and detection in detail.

Error correction:

Suppose that a codeword $c \in C$ is sent and that $y = c + e \in GF(q)^n$ is received, where e denotes the error-vector. Suppose that $wt(e) \leq t$. Compute then the syndrome of the received word

$$\psi(y) = \psi(c) + \psi(e) = \psi(e).$$

Error detection:

Assume that we sent a codeword $c \in C$ and then receive $y = c + e \in GF(q)^n$, with

³The generalization of the *rank nullity Theorem* states, that for a linear map $T : V \rightarrow W$ it holds $\dim(V) = \dim(\text{Im}(T)) + \dim(\ker(T))$.

the error-vector e . Since we can detect up to $2t$ errors, suppose that $wt(e) \leq 2t$. Then we compute

$$\psi(y) = \psi(c) + \psi(e) = 0.$$

Note that the error-vector $e = 0$ if and only if $\psi(e) = 0$ since $wt(e) \leq 2t$.

Syndrome Decoding:

Assume that $y \in GF(q)^n$ is received. The goal now is to find the codeword $c \in C$ which is at shortest distance from y , i.e.

$$d(c, y) \leq d(c', y), \quad \text{for each } c' \in C.$$

Assume that up to $t = d(C)/2$ errors happened in the transmission. For each $e_i \in GF(q)^n$ with $wt(e_i) \leq t$ compute its syndrome $\psi(e_i)$ and create a table. Find in the table the one e_i satisfying $\psi(e_i) = \psi(y)$. Then decode y as $c := y - e_i$.

Chapter 3

Introduction to Finite Geometry

This Section serves as an introduction to finite geometry where we will introduce the general notions of incidence structures, projective spaces and planes.

The material has mainly been taken from the paper of Ball and Weiner [4, Chapters 1 and 2]. Additional theory for this Chapter was used from Assmus and Key [1, Chapter 1], Dembowski [9, Chapter 1.1], Glynn [11, Chapter 0], Liu and Pados [21, pp. 3890-3891] and Cameron [8, Chapter 2].

3.1 Incidence Structures

We are going to introduce the definitions of incidence structures which will be needed for the construction of codes using projective planes.

An *incidence structure* is a triple $S = (\mathcal{P}, \mathcal{L}, \mathcal{I})$, consisting of a set of points \mathcal{P} , a set of lines \mathcal{L} and an incidence relation $\mathcal{I} \subset \mathcal{P} \times \mathcal{L}$ between the points and the lines. In literature, the set \mathcal{L} is sometimes also called the set of blocks instead. We say that a point $p \in \mathcal{P}$ is *incident* to a line $l \in \mathcal{L}$ and we write $p \in l$, if $(p, l) \in \mathcal{I}$. Instead of saying *p is incident to l* we may also say *p lies on l* or *l passes through p*.

The relative position of two points or two lines can be analysed by checking whether they lie on a common line or they contain a common point, respectively. More precisely, two or more points are said to be *collinear* if they are incident to a common line. Similarly, two or more lines are called *concurrent* if they are incident to a single point.

In order to be able to represent an incidence structure S , we define an *incidence graph* of S to be an undirected bipartite graph $\Gamma = (V, E)$, where the set of points \mathcal{P} and the set of lines \mathcal{L} are the vertex partitions V and the elements of the incidence relation \mathcal{I} represent the set of edges E . If all point vertices have degree $t + 1$ and all line vertices have degree $s + 1$, where t and s are positive integers, we call the incidence structure S *regular* and we say that Γ has *order* (s, t) .

From this, we define an *incidence matrix* of a structure S consisting of n points and

m lines to be the $(n \times m)$ matrix $A = (a_{ij})_{\substack{1 \leq i \leq n \\ 1 \leq j \leq m}}$ defined by

$$a_{ij} = \begin{cases} 0, & \text{if } P_i \text{ does not lie on } l_j \\ 1, & \text{if } P_i \text{ lies on } l_j, \end{cases}$$

where P_1, \dots, P_n and l_1, \dots, l_m are some labellings of the n points and the m lines of S , respectively.

Example 3.1.1. Consider an incidence structure consisting of the following sets of points and lines.

$$\begin{aligned} \mathcal{P} &= \{p_1, p_2, p_3, p_4\}, \\ \mathcal{L} &= \{l_1 = \{p_1, p_2\}, l_2 = \{p_2, p_3\}, l_3 = \{p_3, p_4\}, l_4 = \{p_1, p_4\}\}. \end{aligned}$$

Since p_1 is an element of l_1 and l_4 , we say that p_1 is incident to these two lines. This implies that the entries a_{11} and a_{14} of an incidence matrix A must be 1. If we do so for all the points p_i , we obtain the following incidence matrix

$$A = \begin{matrix} & l_1 & l_2 & l_3 & l_4 \\ \begin{matrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{matrix} & \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \end{matrix}.$$

The *dual structure* of an incidence structure $S = (\mathcal{P}, \mathcal{L}, \mathcal{I})$ is defined to be the triple $S^\perp = (\mathcal{P}^\perp, \mathcal{L}^\perp, \mathcal{I}^\perp)$, where $\mathcal{P}^\perp = \mathcal{L}$, $\mathcal{L}^\perp = \mathcal{P}$ and $(l, p) \in \mathcal{I}^\perp$ if and only if $(p, l) \in \mathcal{I}$. An incidence structure S is called *self-dual* if it is isomorphic to its dual S^\perp . This means that the set of points of the structure S is the same as the set of points of S^\perp and analogously for the set of lines. Also it means that incidence is preserved.

Usually we will work with structures that show a certain degree of regularity in their definition. We will call these designs *t-designs*, but let us give a formal definition.

Definition 3.1.2. [1, Definition 1.2.1] An incidence structure $S = (\mathcal{P}, \mathcal{L}, \mathcal{I})$ is called a $t - (v, k, \lambda)$ design, where t, v, k and λ are non-negative integers, if

- i) $|\mathcal{P}| = v$,
- ii) every line $l \in \mathcal{L}$ passes through precisely k points,
- iii) every t distinct points are together incident to precisely λ lines.

A $2 - (v, k, \lambda)$ design is said to be *symmetric*, if the number of points is equal to the number of lines. We will see later, that a two-dimensional projective plane of order q is a symmetric $2 - (q^2 + q + 1, q + 1, 1)$ -design.

3.2 Finite Projective Spaces

Let us first introduce finite projective spaces via the definition of incidence structures.

Definition 3.2.1. [11, Definition 0.3.1] A *finite projective space* is an incidence structure S consisting of a finite number of points and a finite number of lines, that satisfies the following three properties:

- i) Every two distinct points of S are incident to a unique line.
- ii) The number of points on each line of S is greater or equal to three.
- iii) Let l_1 and l_2 be two lines of S intersecting in a unique point P of S . If two points Q and R distinct from P additionally lie on l_1 and another two points T and V distinct from P lie on l_2 , then the line joining Q and T and the line joining R and V intersect in a unique point.

We say a *subset* U of points of a finite projective space S is a *projective subspace*, if for any two distinct points P and Q in U the points lying on the line passing through P and Q are also contained in U .

Example 3.2.2. The empty set is always a subspace of a finite projective space, as well as the set $\{P\}$ consisting of only one point P of the projective space.

The system of all subspaces of a finite projective space S is called a *finite projective geometry*. Note that a finite projective geometry T is again an incidence structure. Indeed, the points of T define the points of the incidence structure and the subspaces of T are the lines of the incidence structure. The incidence relation is then given by the set-theoretical inclusion.

For a finite projective geometry T we can define its dimension and its order.

Definition 3.2.3. [11, Definitions 0.3.6 and 0.3.7] The *dimension* n of a finite projective geometry T is one less than the size of the maximal set of points of T that are linearly independent. We define the *order* q of a finite projective geometry T to be one less than the number of points lying on any line of T .

The following theorem is an important result stating the uniqueness of finite projective geometries of dimension $n \geq 3$.

Theorem 3.2.4. [11, Theorem 0.3.1] *All finite projective geometries of the same dimension $n \geq 3$ and order $q \geq 2$ are isomorphic. Also a finite projective geometry exists only if the order q is a prime power.*

Let us now focus on a finite field $GF(q)$ of q elements, where q is a prime power, and see how we can define a finite projective geometry over $GF(q)$.

Let us denote by $V(n+1, q)$ a vector space of dimension $n+1$ over $GF(q)$. Similar to the definition above, a *projective space*, denoted by $PG(n, q)$, is the projective geometry whose points, lines, planes, \dots , hyperplanes are the subspaces of $V(n+1, q)$

of dimension 1, 2, 3, \dots , n .

Alternatively, we can define the finite projective space $PG(n, q)$ to be the set of equivalence classes $V(n+1, q) \setminus \{0\} / \sim$, where the equivalence relation is defined by $x \sim y$ if there is a non-zero constant $k \in GF(q) \setminus \{0\}$ such that $x = k \cdot y$. Since a finite projective geometry is an incidence structure, we can define the points and lines of $PG(n, q)$ using the equivalence relation.

Hence, two $(n+1)$ -tuples (x_0, \dots, x_n) and (y_0, \dots, y_n) are equivalent if there is a non-zero constant $k \in GF(q) \setminus \{0\}$ such that

$$(x_0, \dots, x_n) = (k \cdot y_0, \dots, k \cdot y_n).$$

The lines of the finite projective space $PG(n, q)$ are defined to be the sets of points

$$\langle a_0, \dots, a_n \rangle = \{[x_0, \dots, x_n] \in PG(n, q) \mid a_0 \cdot x_0, \dots, a_n \cdot x_n = 0\},$$

where $a_i \in GF(q)$ for all $i = 0, \dots, n$ and not all zero.

Remark 3.2.5. Let e_i denote the i^{th} unity-vector, i.e. e_i is the all-zero vector with a one-entry at position i . Then the set $\{e_i \mid i = 0, \dots, n\}$ is a maximal set of linearly independent points of $PG(n, q)$. Hence, by Definition 3.2.3, the finite projective space $PG(n, q)$ has dimension n .

One can explicitly state the number of subspaces of a given dimension of $V(n, q)$ using the following proposition.

Proposition 3.2.6. [4, Proposition 1.2.1] *The number of subspaces of dimension k of the vector space $V = V(n, q)$ is*

$$\begin{bmatrix} n \\ k \end{bmatrix}_q := \frac{(q^n - 1)(q^n - q) \cdots (q^n - q^{k-1})}{(q^k - 1)(q^k - q) \cdots (q^k - q^{k-1})}.$$

Proof. A subspace of dimension k is spanned by k linearly independent vectors of V , say v_1, \dots, v_k . To find the number of all such subsets, we need to find all the possible variations of those vectors v_i for $1 \leq i \leq k$. To start, we take any non-zero vector v_1 . Since V has q^n elements, there are $q^n - 1$ choices for v_1 . Then choose v_2 another non-zero vector in V , which is not in the span of v_1 . Therefore the number of choices for v_2 is $q^n - q$. Proceed in this way for all other vectors v_l for $l \leq k$ and observe, that the number of k linearly independent vectors in V is $(q^n - 1)(q^n - q) \cdots (q^n - q^{k-1})$. Similarly, the number of bases for a subspace of dimension k of V is $(q^k - 1)(q^k - q) \cdots (q^k - q^{k-1})$. The number of subspaces of dimension k of V is then the quotient of the number of all possible combinations of these k vectors and the number of bases for such a subspace of V . Hence, we obtain the desired result. \square

A set \mathcal{S} of a finite projective space $PG(2n-1, q)$ that consists of $q^n + 1$ subspaces of dimension $n-1$ of $PG(2n-1, q)$ which are mutually disjoint is called a *spread set*. Let us define a bijection between the set of subspaces of a projective geometry to itself.

Definition 3.2.7. [11, Definition 0.3.12] We call a bijection ϕ of the set of subspaces of a projective geometry to itself a *collineation*, if it preserves the dimension and the incidence.

Another value needed is a *quadric* of a projective space $PG(n, q)$. We will later in this chapter define a concrete example of a quadric of $PG(2, q)$, called a conic, which is of high importance for the construction presented in Chapter 6.

Definition 3.2.8. [11, Definition 0.3.15] A set of points $\{x_0, \dots, x_n\}$ in a finite projective space $PG(n, q)$ satisfying a quadratic equation $\sum_{i,j=0}^n a_{ij}x_i x_j = 0$, where $a_{ij} \in GF(q)$ for all i and j and not all a_{ij} zero, is called a *quadric* of $PG(n, q)$.

The *associated matrix* with a quadric of $PG(n, q)$ is the $(n + 1) \times (n + 1)$ matrix $B = (b_{ij})$, where $b_{ij} = a_{ij} + a_{ji}$ for all i and j .

The associated matrix B completely determines a quadric of $PG(n, q)$ when q is odd. Furthermore B is a symmetric matrix and it can be used to distinguish two types of quadrics of $PG(n, q)$.

Definition 3.2.9. [11, Definition 0.3.17] A quadric of $PG(n, q)$ with associated matrix B is called *non-degenerate* if it satisfies one of the following properties:

- i) q is odd, or q is even and n odd, and $\det(B) \neq 0$,
- ii) q and n are even, $\text{rk}(B) = n$ and the unique point $x \in PG(n, q)$ satisfying $Bx^\top = 0$ is not an element of the quadric.

A quadric which is not non-degenerate will be called *degenerate*.

According to this property we will make a distinction between the cases where q is even or odd. Concretely, we will mainly focus on the case where q is odd.

3.3 Projective Planes

In Theorem 3.2.4 we have seen that for all $n \geq 3$ there is a unique $PG(n, q)$. For $n = 2$, there can be finite projective geometries different to what we have defined before. These geometries are called projective planes and extend the concept of affine planes. We will present here the definition and properties of a projective plane. Later in this thesis we will mainly focus on the two-dimensional projective plane $PG(2, q)$.

A *projective plane* is an incidence structure Π of points and lines satisfying the following properties:

- (P1) Any two distinct points are incident with exactly one line,
- (P2) Any two distinct lines are incident with exactly one point,
- (P3) There are four points such that no three of them are collinear.

We observe that there is a map σ from a projective plane to its dual plane mapping points to lines and lines to points preserving incidence. The above properties then imply that the dual of a projective plane is a projective plane as well. This means

that (P1)-(P3) are self-dual and whenever a theorem is proven for points in a projective plane then the result automatically holds for lines.

We say that a projective plane is *finite*, if it consists of finitely many points and lines. The following two propositions state the number of points and lines in a finite projective plane of a given order q , as well as the number of points (lines) incident to a line (point).

Proposition 3.3.1. [4, Proposition 1.4.2] *Every point in a projective plane of order q is incident to $q + 1$ lines and every line is incident to $q + 1$ points.*

Proof. Since we are in a projective plane and the properties (P1)-(P3) are self-dual, it suffices to show the result for every point.

Take a point P and a line l in a projective plane, which are not incident. Using the properties (P1) and (P2) we get that the number of lines incident to P is equal to the number of points incident to l . Since a projective space is an incidence structure, there is a bijection between the points on the line l and the lines passing through the point P . So by property (P3), given any two lines, there is a point $Q \neq P$ which is not incident with l . Therefore, the number of lines passing through Q is equal to the number of points lying on l which is equal to the number of lines passing through P . Since P and Q were chosen arbitrarily, we conclude that every point is incident with the same constant number of lines. \square

Proposition 3.3.2. [4, Proposition 1.4.3] *There are $q^2 + q + 1$ points and also $q^2 + q + 1$ lines in a projective plane of order q .*

Proof. Again we will prove the proposition only for points, since the result for lines follows by duality. Let P be a point in the projective plane. Since the order of the projective plane is q , by the previous proposition there are $q + 1$ lines incident to P . In addition, each of these $q + 1$ lines passing through P contain q further points, without repetition. Hence there are totally $q(q + 1) + 1 = q^2 + q + 1$ points in the projective plane of order q . \square

A *configuration* consists of a finite set of points and a finite set of lines, such that each point is incident to the same number of lines and dually each line is incident to the same number of points. Hence, a finite projective plane is a configuration.

Projective planes cannot only be defined using incidence structures. In fact, there are projective planes which can be constructed from a three-dimensional vector space.

To make this more clear, consider a three-dimensional vector space $V(3, q)$ over $GF(q)$. The projective planes, which will be denoted by $PG(2, q)$, that can be constructed from $V(3, q)$, are called *Desarguesian planes*, named after Girard Desargues. Similarly, projective planes that can not be constructed from that vector space are called *non-Desarguesian*. In this thesis we will only focus on Desarguesian planes.

Lemma 3.3.3 ([8]). *The projective space $PG(2, q)$ is a projective plane.*

Proof. Recall that the geometry $PG(2, q)$ contains a set of points consisting of the one-dimensional subspaces of $V(3, q)$ and a sets of lines consisting of the two-dimensional subspaces of $V(3, q)$. Let U and W be two set of lines. Both U and

W are subspaces of dimension 2. This implies that $U + W = V(3, q)$ and so the intersection $U \cap W$ is a subspace of dimension 1. Hence, $U \cap W$ is a point. Therefore, we can conclude that in $PG(2, q)$ every two lines are incident to a (unique) point. Similarly, one can show that every two points in $PG(2, q)$ are incident with exactly one line. We can therefore say that $PG(2, q)$ is a projective plane. \square

Similarly to the parametrization seen in Section 3.2, there is an alternative way to construct a projective plane using equivalence relations. The points of a projective plane $PG(2, q)$ over a finite field $GF(q)$ are identified with the equivalence classes of the set $GF(q)^3 \setminus \{0\}$ modulo the following equivalence relation \sim

$$(x, y, z) \sim \lambda(x, y, z) = (\lambda x, \lambda y, \lambda z), \text{ for } \lambda \in GF(q) \setminus \{0\}.$$

We therefore get that the set of points \mathcal{P} of a projective plane is given by

$$\mathcal{P} = \{[1, b, c] \mid b, c \in GF(q)\} \cup \{[0, 1, c] \mid c \in GF(q)\} \cup \{[0, 0, 1]\}.$$

For a given point $P = [x, y, z] \in \mathcal{P}$, the set of lines \mathcal{L}_P with respect to P is defined to be the set of points

$$\langle a, b, c \rangle := \{[a, b, c] \in PG(2, q) \mid ax + by + cz = 0\}.$$

Example 3.3.4. A famous example of a projective plane is the *Fano plane* $PG(2, 2)$, introduced by the Italian mathematician Gino Fano. It is the finite projective plane consisting of 7 points and 7 lines, which is indeed the smallest possible number of lines and points for a finite projective plane. Using the above definition we obtain the following set of points and lines

$$\begin{aligned} \mathcal{P} &= \{[1, 0, 1], [1, 1, 0], [0, 1, 1], [1, 1, 1], [0, 0, 1], [1, 0, 0], [0, 1, 0]\}, \\ \mathcal{L} &= \{\langle 1, 0, 1 \rangle, \langle 1, 1, 0 \rangle, \langle 0, 1, 1 \rangle, \langle 1, 1, 1 \rangle, \langle 0, 0, 1 \rangle, \langle 1, 0, 0 \rangle, \langle 0, 1, 0 \rangle\}. \end{aligned}$$

This means for instance that the line $\langle 1, 0, 1 \rangle$ is the set of all points $[x, y, z]$ satisfying $a + c = 0$, i.e. all the points for which $x = z$. Hence the line $\langle 1, 0, 1 \rangle$ contains the points $[1, 1, 1]$, $[1, 0, 1]$ and $[0, 1, 0]$.

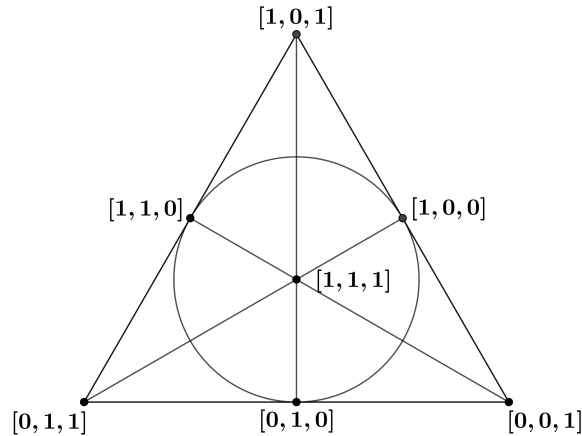


Figure 3.1: The Fano plane $PG(2, 2)$.

We observe that indeed each line passes through three points and each point is incident to three lines. Since in the Fano plane there are three points on each line and similarly three lines passing to each point, the order of the Fano plane is 2.

We can represent the Fano plane with an incidence matrix. Let p_0, \dots, p_6 denote the seven points of \mathcal{P} on the plane and l_0, \dots, l_6 the seven lines of \mathcal{L} . We label the rows with the points and the columns with the lines. Then an incidence matrix is of the form

$$\Gamma = \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \end{array} \begin{array}{ccccccc} l_0 & l_1 & l_2 & l_3 & l_4 & l_5 & l_6 \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \end{array}.$$

Note that this incidence matrix is not unique but it depends on the labelling.

We have seen that $PG(2, q)$ is a projective plane. Using Proposition 3.2.6, we can compute the number of points on a projective line over $GF(q)$ as follows

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix}_q = \frac{(q^2 - 1)}{(q - 1)} = q + 1.$$

This means that there are $q + 1$ points incident to each line and thus the order of the projective plane $PG(2, q)$ is q .

3.4 Arcs and Ovals

Now that we are familiar with the basic definitions of finite projective planes, we can start to define the structure of some objects, such as conics and ovals, in the finite projective plane $PG(2, q)$. We will see that in $PG(2, q)$ all conics are equivalent and that for q odd conics are ovals. The structure of conics will later be needed to construct MDPC-Codes.

Consider a finite projective plane Π of order q . An r -arc is a set of r points of Π with the property that no three distinct points are collinear. The number r of points of an r -arc is dependent on whether the order q is even or odd. The following theorem gives an upper bound on this number r . Since we do not need this result in the course of the following pages, we will state it without proof.

Theorem 3.4.1. [11, Theorem 0.4.1] *Let \mathcal{A} be an r -arc in a finite projective plane of order q , then it holds*

$$\begin{array}{l} \text{if } q \text{ is odd, then } r \leq q + 1, \\ \text{if } q \text{ is even, then } r \leq q + 2. \end{array}$$

We call a $(q+1)$ -arc in a finite projective plane of order q an *oval*. If a line intersects an oval in exactly one point we call this line a *tangent*. In fact there is a unique tangent in every point of an oval. This means, since an oval \mathcal{O} consists of $q+1$ points, there are exactly $q+1$ tangents to an oval \mathcal{O} . If a line meets an oval in two points we call it a *secant*. Finally, lines not intersecting with an oval are called *external lines*.

For a fixed projective plane Π of order q there are some general properties of tangents, secants and external lines of an oval in Π .

Proposition 3.4.2. [11, page 44] *Let \mathcal{O} be an oval in a projective plane of order q . Then \mathcal{O} admits $q+1$ tangents, $\frac{q^2+q}{2}$ secants and $\frac{q^2-q}{2}$ external lines. If additionally q is odd, then the tangents to \mathcal{O} form a dual-oval such that exactly $\frac{q^2+q}{2}$ points lie on exactly two tangents to the oval \mathcal{O} and exactly $\frac{q^2-q}{2}$ points do not lie on any tangent to \mathcal{O} .*

The following figure illustrates this situation.

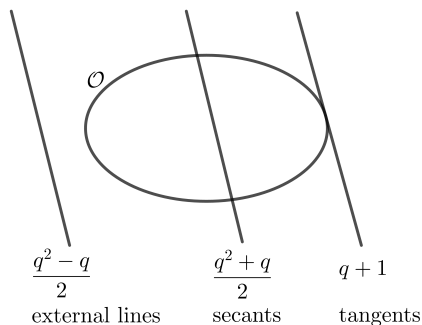


Figure 3.2: External lines, secants and tangents to an oval \mathcal{O} .

Other than ovals we need to introduce another structure in $PG(2, q)$, called a conic. We will give a proper definition of this structure since it plays a central role in this master thesis.

Definition 3.4.3. [11, Definition 0.4.4] A quadric of a projective plane $PG(2, q)$ is called a *conic*.

Hence, by Definition 3.2.8, a conic \mathcal{C} satisfies a quadratic equation. In fact, for some constants $a, b, c, d, e, f \in GF(q)$ the general quadratic equation defining the points of a conic \mathcal{C} is

$$ax^2 + by^2 + cz^2 + dyz + exz + fxy = 0. \quad (3.1)$$

Similarly, we define the *associated matrix* A of a conic to be the homogeneous,

symmetric (3×3) -matrix given by

$$A = \begin{cases} \begin{pmatrix} 2a & f & e \\ f & 2b & d \\ e & d & 2c \end{pmatrix}, & \text{if } q \text{ is odd,} \\ \begin{pmatrix} 0 & f & e \\ f & 0 & d \\ e & d & 0 \end{pmatrix}, & \text{if } q \text{ is even.} \end{cases}$$

Conics of $PG(2, q)$ are distinguished in two types, the non-degenerate and the degenerate ones. Recall from Definition 3.2.9 that a conic is said to be non-degenerate if the associated matrix A is non-singular, when q is odd, or if the associated matrix has rank 1 and the unique point $x \in PG(2, q)$ satisfying $Ax^\top = 0$ is not an element of the conic.

We will focus on the case when q is odd since we want to work with non-degenerate conics. As we will see in Section 4.1, non-degenerate conics mimic the properties of lines in $PG(2, q)$. Also their oval shape, see Table 3.1 below, provide some nice properties which will also be discussed in Chapter 4.

If we fix q to be odd, with a change of variables in the general quadratic equation 3.1 defining a conic, we are able to use the quadratic equation

$$Y^2 = XZ \quad \text{or} \quad X^2 + Y^2 - Z^2 = 0,$$

which is more convenient and customary.

With this definition one gets that every non-degenerate conic of the Desarguesian plane $PG(2, q)$ is an oval. In 1955, Segre [29] proved the converse statement for q odd, which in 1949 was conjectured by Järnefelt and Kustaanheimo [19]. Several different proofs have been presented, the original proof can be found in [29], another more involved one in [4].

Theorem 3.4.4 (Segre, [29]). *If $\text{char}(GF(q)) \neq 2$, every oval of $PG(2, q)$ is a conic (i.e., can be represented by an equation of second degree).*

Indeed the converse is true when the conic is non-degenerate, which is the following theorem:

Theorem 3.4.5. [11, Theorem 0.4.5] *A conic of a Desarguesian plane of order q is an oval if and only if it is non-degenerate.*

In Segre's lectures of modern geometry from 1961 we find that there are in fact three types of degenerate conics (i.e. conics that are not non-degenerate) and only one type of non-degenerate conics, the ovals (concern [30] for detailed definitions of the particular types of conics). The following table shows the four different types of conics (see [11], p. 86) with the corresponding diagram and number of points contained.

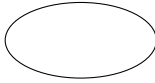
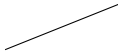
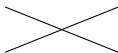
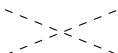
Type of conic	Number of points in conic	Diagram
non-degenerate	$q + 1$	
repeated line	$q + 1$	
two real lines	$2q + 1$	
two imaginary lines (intersecting in a real point)	1	

Table 3.1: The four distinct types of conics in $PG(2, q)$.

3.5 Codes from Projective Planes

In this section we are going to show the relation between coding theory and finite geometry. Indeed, the idea of constructing error-correcting codes using projective planes came up in the late 1950s. Prange studied the construction of error-correcting codes using projective planes of order 8 (see [27] for more details). Also Rudolph investigated the use of projective planes for the construction of error-correcting codes. He actually was the first person who showed that codes of least redundancy are achieved with projective planes (see [28]).

One of the most basic example showing the relation is given by the projective plane of order 2 (i.e. the Fano plane) and the $[7, 4, 3]$ binary Hamming code. In Example 3.3.4 we have given a possible incidence matrix Γ of the Fano plane. If we define $C = \ker(\Gamma)$, then Γ is a parity-check matrix of the code C . From the parity-check matrix we obtain that C is a $[7, 4, 3]$ binary Hamming code.

For the general definition of a code from a projective plane we need a projective plane Π of order n and a field F . Then we define the code $C_F(\Pi)$ to be the row-space of an incidence matrix H of Π over the field F .

From this it follows that the dual code is the linear code

$$C_F(\Pi)^\perp = \ker(H).$$

If the field F consists of p elements we denote the code $C_F(\Pi)$ by $C_p(\Pi)$.

If the number of elements p of a field F is a prime number dividing the order q of the projective plane, the code $C_p(\Pi)$ is an $[q^2 + q + 1, k, q + 1]_2$ -linear code (see [1], p.204). Clearly, the code length is given by the number of lines in a projective plane, which by Proposition 3.3.2, is $q^2 + q + 1$, since the plane is of order q . Usually it is quite easy to determine the dimension of a code and rather difficult to find the minimum distance. In the case here it is the other way around, since the dimension is highly dependent on the structure of the projective plane Π . Therefore, we will denote the dimension of the code only by k without giving a concrete value.

We are mainly interested in the dual code $C_p(\Pi)^\perp$. Assmus and Key gave an estimate on the minimum distance of this code $C_p(\Pi)^\perp$, which is the following corollary.

Corollary 3.5.1. [1, Corollary 6.3.1] *If Π is a projective plane of order q and p is a prime number dividing q , then the minimum weight of $C_p(\Pi)^\perp$ is at least $q + 2$. If the minimum weight is exactly $q + 2$ then, necessarily, $p = 2$ and q is even.*

This provides a lower bound on the minimum distance for the code $C_p(\Pi)^\perp$.

Other lower bounds on the minimum distance for arbitrary binary linear codes were provided by Robert Michael Tanner in 2001 using a bipartite code constraint graph ([36]). Given a parity-check matrix $H = (h_{ij})_{\substack{1 \leq i \leq r \\ 1 \leq j \leq n}}$ of size $r \times n$ of a binary linear code C of length n , we can deduce a bipartite graph $G = (V, E)$, where the set of vertices (or nodes) V is the union of the set of parity nodes and the set of check nodes. The set of edges is denoted by E . There is an edge between a parity node i and a check node j if and only if $h_{ij} = 1$ in the parity-check matrix H . Note that this graph representation of the code is not unique since also the parity-check matrix is not unique, and the bounds are dependent on that representation.

Let A denote the real matrix product HH^\top having distinct ordered eigenvalues $\lambda_1 > \dots > \lambda_s$. Since the matrix A is symmetric, the eigenvalues are all real-valued. The first bound shows a relation between the bit nodes in a minimum distance codeword of a binary linear code.

Theorem 3.5.2. [36, Theorem 3.1] *Let G be a bipartite graph deduced from an $(r \times n)$ parity-check matrix H of a binary linear code C of length n . If G is a regular connected graph with n bit nodes all of degree m , and r parity nodes all of degree j , the minimum distance $d(C)$ of the code satisfies*

$$d(C) \geq \frac{n(2m - \lambda_2)}{mj - \lambda_2}.$$

Note that the bound becomes meaningless if $\lambda_2 \geq 2m$.

Another bound, which was found by Tanner, considers the connectivity between parity nodes.

Theorem 3.5.3. [36, Theorem 4.1] *Let G be a bipartite graph deduced from an $(r \times n)$ parity-check matrix H of a binary linear code C of length n . If G is a regular connected graph with n bit nodes all of degree m , and r parity nodes all of degree j , the minimum distance $d(C)$ of the code satisfies*

$$d(C) \geq \frac{n(2m + j - 2 - \lambda_2)}{j(mj - \lambda_2)}.$$

If here $\lambda_2 \geq 2m$, then this bound can still be meaningful, whereas the above bound is not. However, this does not imply that this bound is always stronger.

Chapter 4

Projective Bundles

The aim of this Section is to discuss the construction and existence of projective bundles which are a collection of non-degenerate conics of the Desarguesian projective plane $PG(2, q)$ of order q which are mutually intersecting in a unique point. We will mainly focus on the case where q is odd. Projective bundles will play an important role in our construction of MDPC-codes.

The material for this Chapter is mainly taken from Chapter 0 and 1 of Glynn's Ph.D. Thesis [11], who studied projective bundles, which he called *packings* of k -arcs. He studied three different types of those packings and the connection between conics in the Desarguesian plane $PG(2, q)$ and points of the projective plane $PG(5, q)$. In fact he showed that those three types of projective bundles correspond to certain planes of $PG(5, q)$. Additional material has been taken *inter alia* from Baker, Brown, Ebert and Fisher [2].

4.1 Definitions and Basic Results

Consider the Desarguesian projective plane $PG(2, q)$ of odd order q coordinatized over $GF(q)$ in the classical way, which is the construction over a three-dimensional vector space over $GF(q)$ presented in Section 3.3. Recall from Section 3.4, that a conic is a set \mathcal{C} of $q + 1$ points of $PG(2, q)$ satisfying a quadratic equation and that \mathcal{C} is called non-degenerate if its associated matrix has a non-zero determinant.

Glynn introduced projective bundles as packings of $(q + 1)$ -arcs giving the following definition.

Definition 4.1.1. [11, Definition 1.1.4] A *packing of $(q + 1)$ -arcs* of a projective plane Π of order q is a set \mathcal{C} of $(q + 1)$ -arcs of Π satisfying the following conditions:

- i) $|\mathcal{C}| = q^2 + q + 1$.
- ii) For all $K, L \in \mathcal{C}, K \neq L \implies |K \cap L| = 1$.
- iii) There is a unique $(q + 1)$ -arc of \mathcal{C} containing any two distinct points of Π .
- iv) There are $q + 1$ elements of \mathcal{C} which are passing through any particular point of Π .

Earlier we have seen that $(q + 1)$ -arcs are called ovals and from Segres theorem, Theorem 3.4.4, we know that if q is odd every oval in the Desarguesian plane $PG(2, q)$ is a non-degenerate conic. Using this we can rewrite Glynn's definition of a packing of $(q + 1)$ -arcs as the following definition of a projective bundle.

Definition 4.1.2 ([2]). A *projective bundle* is a collection of $q^2 + q + 1$ non-degenerate conics of $PG(2, q)$ for q odd, satisfying that every two distinct conics are mutually intersecting in a single point.

One can easily see that projective bundles represent a projective plane of order q . In that setting one can consider the conics to be the lines of $PG(2, q)$. Hence, we can say that in $PG(2, q)$ the set of points \mathcal{P} is represented in the set of lines \mathcal{L} by a projective bundle. We recall furthermore, that there is a bijection f from the projective plane $PG(2, q)$ to its dual mapping the points to the lines and vice versa preserving incidences. In addition, if A is an incidence matrix corresponding to $PG(2, q)$ and B an incidence matrix of the dual of $PG(2, q)$, there is a bijection f mapping the i^{th} row of A to the i^{th} column of B .

Let $P \in \mathcal{P}$ be a point of the Desarguesian plane of order q and $l \in \mathcal{L}$ a line. We define C_P with respect to a point P to be the pre-image of all lines passing through P under the bijection f . Similarly, define C_l with respect to a line will be defined as the image of the set of points lying on the line l under f , i.e.

$$\begin{aligned} C_P &= f^{-1}(\text{lines of } PG(2, q) \text{ through } P), \\ C_l &= f(\text{points of } PG(2, q) \text{ lying on } l). \end{aligned}$$

The row of A corresponding to point P gives the intersection numbers of lines of $PG(2, q)$ with C_P . Since this row is made up of zero- and one-entries, it follows that C_P is a $(q + 1)$ -arc in $PG(2, q)$. Denote by $A_{(P,l)}$ the entry of an incidence matrix A of $PG(2, q)$ at the point P and line l . Then we can deduce

$$\begin{aligned} A_{(P,l)} = 1 &\iff \text{Point } P \text{ is on line } l, \\ &\iff l \text{ is tangent to } C_P, \\ &\iff P \text{ is on one line of } C_l. \end{aligned}$$

Additionally, if we have two distinct points P and Q , we have that the conics C_P and C_Q , defined as above, satisfy

$$C_P \cap C_Q = f^{-1}(\langle P, Q \rangle).$$

Hence, the conics C_P and C_Q intersect in a unique common point and by duality they have a unique common tangent as shown in the following figure.

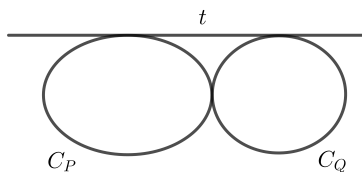


Figure 4.1: Intersection of two distinct conics.

Another quantity needed is the notion of *pencil of conics*. Assume we have two distinct conics represented by the two independent quadratic equations $C_1 = 0$ and $C_2 = 0$. Then for every $\lambda \in GF(q)$ the equation $C_1 + \lambda C_2 = 0$ also represents a conic.

Definition 4.1.3 ([11]). The set \mathcal{P} consisting of the conics represented by $C_2 = 0$ and $C_1 + \lambda C_2 = 0$, for all $\lambda \in GF(q)$, is called a *pencil of $q + 1$ conics* generated by $C_1 = 0$ and $C_2 = 0$.

Previously, in Section 3.4, we have seen that there are 4 types of conics, namely non-degenerate conics, repeated lines, line pairs and two imaginary lines intersecting in one point. Hence, there are various types of pencils of conics that contain non-degenerate conics. In fact, Glynn has shown that there are exactly nine different types of pencils containing non-degenerate conics (for more details see [11, table 1.2.2]). Since we are only interested in a packing of conics containing only non-degenerate conics, we also are interested in a pencil containing only non-degenerate conics. It turns out, that there is indeed only one type of pencil that consists of $q + 1$ conics which are all non-degenerate. In addition, every two distinct non-degenerate conics in this type of pencil intersect in exactly one point.

4.2 Existence of Projective Bundles

As mentioned at the beginning of this chapter, we will only focus on the conditions for projective bundles to exist in the Desarguesian plane $PG(2, q)$ of odd order q .

We already know how to represent a non-degenerate conic by a (3×3) -matrix (see Section 3.4). The goal now is to find a matrix-representation of a projective bundle. The following statement gives such a characterization of matrices being associated to projective bundles of the Desarguesian plane.

Theorem 4.2.1. [11, Theorem 1.2.1] *Let $PG(2, q)$ be coordinatized over $GF(q)$ in the classical way, where q is an odd prime power. The matrices A_1, \dots, A_{q^2+q+1} are the associated matrices of a projective bundle in $PG(2, q)$ if and only if*

- i) each matrix A_i is a non-singular symmetric, (3×3) -matrix over $GF(q)$ for all $i = 1, \dots, q^2 + q + 1$, and*
- ii) $A_i A_j^{-1}$ is fixed point free, meaning that its characteristic polynomial is irreducible over $GF(q)$ for all $i \neq j$ with $i, j = 1, \dots, q^2 + q + 1$.*

Proof. Let us first assume that A_1, \dots, A_{q^2+q+1} are the matrices of a projective bundle in $PG(2, q)$. That means that each A_i is an associated matrix to the conic C_i of the packing. By the definition of an associated matrix to a non-degenerate conic (see Section 3.4 for definition), all A_i satisfy property *i*) of the theorem.

For the second property, we make use of pencils of conics. For this, let C_i and C_j be the conics associated to the matrices A_i and A_j , where $i \neq j$. Since both belong to a projective bundle and by the bijection discussed in Section 4.1, they need to have a unique intersection point as well as a unique common tangent not passing through

this common point. Therefore, the pencil generated by $C_i = 0$ and $C_j = 0$ must only contain non-degenerate conics. If this was not the case, two distinct non-degenerate conics of the pencil would have either one line of the line pair in the pencil or the unique repeated line of the pencil as a common tangent, which by construction must also pass through the common point.

Since the pencil contains the conics represented by $C_j = 0$ and $C_i + \lambda C_j = 0$ for any $\lambda \in GF(q)$, the matrices of these conics are A_j and $A_i + \lambda A_j$. Furthermore, we have that all the conics of the pencil are non-degenerate which, by definition, implies that A_j and $A_i + \lambda A_j$ are non-singular for every $\lambda \in GF(q)$. Hence,

$$\det(A_i + \lambda A_j) \neq 0, \text{ for all } \lambda \in GF(q),$$

or equivalently

$$\det(A_i A_j^{-1} + \lambda \mathbb{I}_3) \neq 0, \text{ for all } \lambda \in GF(q),$$

where \mathbb{I}_3 is the (3×3) identity-matrix.

Therefore, we conclude that the matrix $A_i A_j^{-1}$ has no eigenvalues and thus, by Cayley-Hamilton, satisfies an irreducible cubic equation.

For the converse statement, assume that A_1, \dots, A_{q^2+q+1} satisfy the properties *i*) and *ii*) of the theorem. We need to check that they represent non-degenerate conics. Let A_i and A_j be two of these matrices which are distinct. Since $A_i A_j^{-1}$ is non-singular, the matrices A_i and A_j correspond to non-degenerate conics say C_i and C_j , by the argument used above. Since q is odd, there exist dual conics represented by the matrices A_i^{-1} and A_j^{-1} in dual coordinates. Therefore, since $A_i A_j^{-1}$ satisfies an irreducible cubic equation, also $(A_i^{-1}) (A_j^{-1})^{-1} = A_i^{-1} A_j$ satisfies an irreducible cubic equation, which implies that A_i and A_j have a unique common tangent not passing through the common point. Hence, the distinct conics C_i and C_j represented by A_i and A_j , respectively, intersect as shown in Figure 4.1.

Let C be the set of the conics corresponding to the matrices A_1, \dots, A_{q^2+q+1} and let P be a point of $PG(2, q)$. Since we are in a projective plane, there are $q + 1$ lines through P all of them being tangent to at most one conic of C passing through P . Therefore, we can say that there are at most $q + 1$ conics of C passing through P . Since $PG(2, q)$ consists of $q^2 + q + 1$ points and C consists of $q^2 + q + 1$ conics all having $q + 1$ points, there are exactly $q + 1$ conics of C through each point of $PG(2, q)$. Hence, we conclude that C is a projective plane of order q and thus it is a projective bundle in $PG(2, q)$. \square

Recall from Section 3.2 that a spread set of a finite projective space $PG(2n - 1, q)$ is a set of $q^n + 1$ subspaces of $PG(2n - 1, q)$ of dimension $n - 1$ which are mutually disjoint. Hence, a spread set of $PG(5, q)$ is a set of $q^3 + 1$ subspaces of dimension 1 that are mutually disjoint.

However, applying the above theorem to a spread of $PG(5, q)$, we observe that it corresponds to a projective bundle in $PG(2, q)$ for odd order q . Generally, a spread set is closed under addition and can be viewed as vector space of dimension n over $GF(q)$. This implies that it is also closed under multiplication by elements of $GF(q)$ and that it corresponds to a set of $(n \times n)$ -matrices over $GF(q)$ which correspond

to a semi-field of order q^n . Hence, it has a basis of n matrices in S . Therefore, the previous theorem has the following important corollary.

Corollary 4.2.2. [11, Corollary 1.2.1] *For every commutative semi-field of dimension 3 over $GF(q)$ there exists a projective bundle in $PG(2, q)$.*

We observe that $GF(q^3)$ is the unique field of dimension 3 over $GF(q)$, which is clearly also a commutative semi-field. Hence, there is always a projective bundle in $PG(2, q)$ for q odd.

4.3 Geometrical Representation in $PG(5, q)$

From the above discussion we conclude that there is a connection of projective bundles in $PG(2, q)$ and spread sets in $PG(5, q)$. In the following we will see that non-degenerate conics and projective bundles are related to certain points and planes in $PG(5, q)$, respectively. To see this, we need to introduce the Veronese surface and its properties and relationship to conics in the Desarguesian plane.

Definition 4.3.1. [11, Definition 1.2.2] The *Veronese surface* \mathcal{V} is a surface in the five-dimensional projective plane $PG(5, q)$ of odd order q consisting of $q^2 + q + 1$ points that satisfy the parametric equations

$$[x_0, x_1, x_2, x_3, x_4, x_5] = [a_0^2, a_1^2, a_2^2, a_1a_2, a_0a_2, a_0a_1],$$

where $a_0, a_1, a_2 \in GF(q)$ not all zero. Furthermore, no three points of \mathcal{V} are collinear.

More precisely, \mathcal{V} can be represented by the image of the map

$$\begin{aligned} \nu : PG(2, q) &\longrightarrow PG(5, q) \\ [x, y, z] &\longmapsto [x^2, y^2, z^2, yz, xz, xy]. \end{aligned}$$

Let us identify the points $[x_0, x_1, x_2, x_3, x_4, x_5]$ of $PG(5, q)$ with the symmetric, homogeneous (3×3) -matrix over $GF(q)$

$$\begin{pmatrix} x_0 & x_5 & x_4 \\ x_5 & x_1 & x_3 \\ x_4 & x_3 & x_2 \end{pmatrix}. \quad (4.1)$$

Therefore, the Veronese surface can be characterized using points $[x, y, z]$ of $PG(2, q)$ by matrices

$$\begin{pmatrix} x^2 & xy & xz \\ xy & y^2 & yz \\ xz & yz & z^2 \end{pmatrix}.$$

Remember from Section 3.4 and from Display 3.1, that non-degenerate conics in the Desarguesian plane satisfy a general quadratic equation $ax^2 + by^2 + cz^2 + dyz + exz + fxy = 0$ and that their associated matrix is of the form

$$A = \begin{pmatrix} 2a & f & e \\ f & 2b & d \\ e & d & 2c \end{pmatrix}.$$

Therefore, there is a natural connection of the Veronese surface in $PG(5, q)$ and the conics in $PG(2, q)$. Indeed, the homogeneous, symmetric (3×3) -matrix over $GF(q)$ identifying the points of $PG(5, q)$ correspond exactly to the conics of $PG(2, q)$. Hence, we can say that there is a one-to-one correspondence between the points of the five-dimensional projective plane with the conics of the Desarguesian plane.

Related to the Veronese surface \mathcal{V} in $PG(5, q)$ there are two sets of planes which we need to mention.

Definition 4.3.2. [11, page 98 ff.] Let \mathcal{V} be a Veronese surface in $PG(5, q)$ and q an odd prime power. A set \mathcal{C} of $q^2 + q + 1$ planes of $PG(5, q)$ is called a set of *conic planes* of \mathcal{V} if it satisfies the following properties:

- i) The points in each $A \in \mathcal{C}$ correspond to the degenerate conics of $PG(2, q)$.
- ii) For every plane $A \in \mathcal{C}$, the intersection with the Veronese surface \mathcal{V} is a non-degenerate conic of the Desarguesian plane.
- iii) For every plane $A \in \mathcal{C}$, the configuration formed by the intersection $A \cap \mathcal{V}$ is isomorphic to $PG(2, q)$, hence a projective plane of odd order q .

Furthermore, a set \mathcal{T} of $q^2 + q + 1$ planes of $PG(5, q)$ is called a set of *tangent planes* of \mathcal{V} if the following properties are fulfilled:

- i) For every point P of \mathcal{V} there is a unique plane $T \in \mathcal{T}$ that intersects \mathcal{V} only in P , i.e. $T \cap \mathcal{V} = P$.
- ii) Let T_1 and T_2 be two distinct planes of \mathcal{T} . Then they intersect in a unique point. Additionally, there are at most two distinct tangent planes passing through the same point.

It can be shown that for each Veronese surface \mathcal{V} there exist a set \mathcal{C} of conic planes and a set \mathcal{T} of tangent planes. But since we do not want to go into details about the properties of the Veronese surface, we will not prove this fact.

Knowing the connection between conics of $PG(2, q)$ and the points of $PG(5, q)$, we want to find an object of $PG(5, q)$ corresponding to projective bundles in $PG(2, q)$. The next theorem provides exactly such a correspondence.

Theorem 4.3.3. [11, Theorem 1.2.2] *There is a one-to-one correspondence between the projective bundles in the Desarguesian plane and the sets \mathcal{P} of $q^2 + q + 1$ points in $PG(5, q)$ satisfying that the line connecting two distinct points $P, Q \in \mathcal{P}$ does not intersect the set of points in $PG(5, q)$ corresponding to a degenerate conic.*

Proof. From Theorem 4.2.1 we get that a projective bundle exists in $PG(2, q)$ if and only if there exist $q^2 + q + 1$ non-singular, symmetric (3×3) -matrices A_i such that $A_i A_j^{-1}$ satisfies an irreducible cubic equation. This is if and only if the pencil corresponding to each pair of conics of the projective bundle consists only of non-degenerate conics. Hence, such pencils correspond exactly to the lines of $PG(5, q)$ not passing through points which belong to degenerate conics. Therefore, we have found a correspondence of projective bundles in $PG(2, q)$ with a set of points in $PG(5, q)$ and the desired result follows. \square

Hence, planes of the five-dimensional projective plane $PG(5, q)$ of odd order q not intersecting degenerate conics are examples of such sets of points described in the above theorem.

4.4 Classification

There is indeed a one-to-one correspondence between any set of $q^2 + q + 1$ points of $PG(5, q)$ not intersecting in a degenerate conic of $PG(2, q)$ and the projective bundles of $PG(2, q)$. Heretofore we only know that there exists such a correspondence but we do not know which sets of points satisfy this. This section is meant to classify all the planes in $PG(5, q)$ that correspond to a projective bundle in the Desarguesian plane $PG(2, q)$.

Let us first introduce some notation to simplify the terminology. We will denote by Ω the set of points of $PG(5, q)$ corresponding to degenerate conics of $PG(2, q)$. As we have seen in Table 3.1, there are three different types of degenerate conics, namely repeated lines, two real lines or two imaginary lines intersecting in a real point. We will denote the sets of points of $PG(5, q)$ corresponding to one of these three types of degenerate conics by \mathcal{V} , Ω^e and Ω^i , respectively, where Ω^e is called the set of *external points* of Ω and Ω^i the set of *internal points*. Note that \mathcal{V} denotes a Veronese surface. Thus, we have $\Omega = \mathcal{V} \cup \Omega^e \cup \Omega^i$.

Secondly, consider the symmetric, homogeneous (3×3) -matrix identifying a Veronese surface, given in Definition 4.3.1. Indeed, such a matrix is of rank 1 if and only if it is an element of the Veronese surface \mathcal{V} . This means that each point of \mathcal{V} can be represented by a symmetric, homogeneous (3×3) -matrix of rank 1. The points of Ω^e and Ω^i can be represented by such a matrix which is of rank 2.

Recall from Section 4.2 and Section 4.3, that any semi-field of dimension 3 over $GF(q)$ characterized by symmetric (3×3) -matrices over $GF(q)$ corresponds to a plane of $PG(5, q)$ which does not intersect Ω , i.e. a plane which corresponds to a non-degenerate conic in $PG(2, q)$.

In the following, whenever we speak of a point P of $PG(5, q)$ we will always work it with its representation as a (3×3) homogeneous symmetric matrix, shown in Display (4.1). This representation enables us to define the product of two points P and Q in $PG(5, q)$ by the matrix product of their associated matrices. In addition, if the determinant of the matrix is non-zero, we are able to define the inverse P^{-1} of a point $P \in PG(5, q)$ via the inverse of the matrix representing P .

Another field of $PG(5, q)$ not intersecting Ω is the field plane, given by the following definition:

Definition 4.4.1. [11, Definition 1.2.3] A *field plane* is a plane $\Pi = \langle P, Q, PQ^{-1}P \rangle$ of $PG(5, q)$ spanned by the points P, Q and $PQ^{-1}P$, where P and Q are two distinct points of $PG(5, q)$ corresponding to a non-degenerate conic in $PG(2, q)$ and PQ^{-1} satisfies an irreducible cubic equation.

Note that the point Q in $PG(5, q)$ mentioned in this definition indeed admits an inverse, since it corresponds to a non-degenerate conic. A non-degenerate conic, by Definition 3.2.9, is represented by a non-singular associated matrix. Hence this matrix is invertible and its inverse represents the inverse Q^{-1} .

The following is an important property of a field plane.

Theorem 4.4.2. [11, Theorem 1.2.3] *The field planes of $PG(5, q)$ with respect to the Veronese surface \mathcal{V} are permuted amongst themselves by the collineations that fix \mathcal{V} .*

Proof. Consider a field plane Π of $PG(5, q)$ with respect to the Veronese surface \mathcal{V} . By the above definition the field plane is of the form $\Pi = \langle P, Q, PQ^{-1}P \rangle$, where P and Q are two distinct points of $PG(5, q)$ satisfying the properties to span a field plane. Let σ be a collineation of $PG(5, q)$ that fixes \mathcal{V} . This implies that only the planes intersecting \mathcal{V} in a non-degenerate conic are permuted by σ . As we have seen, these planes are exactly the $q^2 + q + 1$ conic planes of \mathcal{V} . Since, by Definition 4.3.2, the configuration formed by the intersections of conic planes and \mathcal{V} is isomorphic to $PG(2, q)$, ϕ corresponds exactly to a collineation of $PG(2, q)$. Hence, a general point of \mathcal{V} is

$$X = \begin{pmatrix} a_0^2 & a_0a_1 & a_0a_2 \\ a_0a_1 & a_1^2 & a_1a_2 \\ a_0a_2 & a_1a_2 & a_2^2 \end{pmatrix} = \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} \cdot (a_0 \ a_1 \ a_2),$$

where $[a_0, a_1, a_2] \in PGF(2, q)$. Also, a general collineation of $PG(2, q)$ is of the form

$$[a_0, a_1, a_2] \mapsto (a_0, a_1, a_2)^\sigma G^\top,$$

where G is a non-singular (3×3) -matrix over $GF(q)$ and σ is an automorphism of $GF(q)$. Therefore, it follows that ϕ is of the form

$$\begin{aligned} \phi : PG(5, q) &\longrightarrow PG(5, q) \\ X &\longmapsto GX^\sigma G^\top. \end{aligned}$$

We need to check that Π transformed under ϕ is still a field plane. If we do so, we obtain that Π has been transformed to

$$\phi(\Pi) = \langle GP^\sigma G^\top, GQ^\sigma G^\top, G(PQ^{-1}P)^\sigma G^\top \rangle.$$

Consider the last point $G(PQ^{-1}P)^\sigma G^\top$ spanning $\phi(\Pi)$. Using simple computations we get

$$\begin{aligned} G(PQ^{-1}P)^\sigma G^\top &= GP^\sigma(Q^\sigma)^{-1}P^\sigma G^\top \\ &= GP^\sigma G^\top \cdot (G^\top)^{-1}(Q^\sigma)^{-1}G^{-1} \cdot GP^\sigma G^\top \\ &= GP^\sigma(Q^\sigma)^{-1}G^{-1} \cdot GP^\sigma G^\top \\ &= G(PQ^{-1})^\sigma G^{-1} \cdot GP^\sigma G^\top, \end{aligned}$$

which satisfies an irreducible cubic equation since PQ^{-1} does, by Definition 4.4.1 of a field plane. Hence $\phi(\Pi) = \langle GP^\sigma G^\top, GQ^\sigma G^\top, G(PQ^{-1})^\sigma G^{-1} \cdot GP^\sigma G^\top \rangle$ is a field plane too and the result follows. \square

From this theorem we can therefore deduce that a field plane Π does not intersect the set Ω of points of $PG(5, q)$ corresponding to degenerate conics in $PG(2, q)$, which means $\Pi \cap \Omega = \emptyset$.

Recall, from Theorem 4.3.3, that a projective bundle corresponds to a set of $q^2 + q + 1$ points of $PG(5, q) \setminus \Omega$ with the property that each line passing through two points of this set does not intersect Ω . By duality arguments we can deduce that there is a dual projective bundle corresponding to a set of tangents to the projective bundle. Therefore, we have the following classification of such a set \mathcal{P} of $q^2 + q + 1$ points.

Lemma 4.4.3. [11, Lemma 1.2.3] *A set \mathcal{P} of $q^2 + q + 1$ points of $PG(5, q)$ not intersecting Ω satisfies the properties of Theorem 4.3.3 if and only if its inverse $\mathcal{P}^{-1} = \{P^{-1} \mid P \in \mathcal{P}\}$ satisfies the same properties.*

Proof. The statement is an immediate consequence of the discussions above. \square

An example of such a set of points would be a field plane, since it consists of $q^2 + q + 1$ points of the five-dimensional projective plane of order q and is not intersecting Ω .

Thus, the natural question that arises is when such a set of points of $PG(5, q)$ not intersecting Ω is in fact a field plane. The following theorem gives a characterization of field planes in the five-dimensional projective plane of order q .

Theorem 4.4.4. [11, Theorem 1.2.4] *A plane Π of $PG(5, q)$ not intersecting Ω is a field plane if and only if $\Pi^{-1} = \{P^{-1} \mid P \in \Pi\}$ is a field plane of $PG(5, q)$.*

Proof. Firstly, assume that $\Pi = \langle P, Q, PQ^{-1}P \rangle$ is a field plane. Since $P, Q, PQ^{-1}P \in \Pi$, Π^{-1} contains the field $\langle P^{-1}, Q^{-1}, P^{-1}QP^{-1} \rangle$ which, by definition, is a field plane. Let in addition $X \in \langle I, PQ^{-1}, (PQ^{-1})^2 \rangle$ be a generator of $GF(q)[PQ^{-1}]$, then the field plane Π can be rewritten as

$$\Pi = \{X^n P \mid n = 1, \dots, q^2 + q + 1\}.$$

Hence, we obtain

$$\Pi^{-1} = \{P^{-1}X^{-n} \mid n = 1, \dots, q^2 + q + 1\},$$

which is contained in a plane $\langle P^{-1}, P^{-1}X, P^{-1}X^2 \rangle$ not intersecting Ω . Therefore we conclude that $\Pi^{-1} = \langle P^{-1}, Q^{-1}, P^{-1}QP^{-1} \rangle$ is a field plane of $PG(5, q)$ not intersecting Ω .

For the converse statement, assume that both Π and Π^{-1} are planes of $PG(5, q)$ not intersecting Ω . Let A and B be two distinct points of Π . Then, by definition, A^{-1}, B^{-1} and $(A + \lambda B)^{-1}$ are points of Π^{-1} for any non-zero $\lambda \in GF(q)$. Since Π is a plane that does not intersect Ω , the line $\langle A, B \rangle$ passing through A and B does not intersect Ω either. Hence, the point $A^{-1}B$ satisfies an irreducible cubic equation, say $f(x) = x^3 + rx^2 + sx + t = 0$. Hence, the points A^{-1}, B^{-1} and $A^{-1}BA^{-1}$ generate a field plane $\langle A^{-1}, B^{-1}, A^{-1}BA^{-1} \rangle$. Furthermore, we have

$$\begin{aligned} (A + \lambda B)^{-1} &= (B^{-1}A + \lambda \mathbb{I})^{-1}B^{-1} \\ &= (a\mathbb{I} + b \cdot A^{-1}B + c \cdot A^{-1}BA^{-1}B)B^{-1}, \end{aligned}$$

and hence,

$$(B^{-1}A + \lambda\mathbb{I})^{-1} = (a\mathbb{I} + b \cdot A^{-1}B + c \cdot A^{-1}BA^{-1}B)$$

for some constants $a, b, c \in GF(q)$.

Solving this equation yields

$$\frac{a}{c} = \lambda t, \quad \frac{b}{c} = r - \frac{1}{\lambda} \quad \text{and} \quad \frac{1}{c} = \lambda^2 f\left(\frac{-1}{\lambda}\right).$$

Since λ is non-zero and the function f is irreducible, c is non-zero too. Therefore, and because A^{-1}, B^{-1} and $A^{-1}BA^{-1}$ generate a field plane, Π^{-1} is a field plane. Hence, by the first part of the proof we conclude that Π must be a field plane as well. \square

Having now fully characterized the field planes of $PG(5, q)$ and knowing that they correspond to projective bundles of $PG(2, q)$ for any odd prime power q , we want to figure out whether or not there exist any other planes in $PG(5, q)$ not intersecting a degenerate conic of the Desarguesian plane $PG(2, q)$. The following theorem classifies all these planes in $PG(5, q)$, for q odd. It is the main result for projective bundles in $PG(2, q)$.

Theorem 4.4.5. [11, Theorem 1.2.6] *Suppose q is an odd prime power. Then through each line of $PG(5, q)$ which does not intersect Ω there are exactly two planes not intersecting Ω . One is the field plane and the other will be called a semi-field plane.*

Glynn's proof of this Theorem 4.4.5 does not only verify the statement but in addition it provides a complete construction of the two desired planes. The proof makes use of a result due to Dickson and Segre, stating that a cubic curve in $PG(2, q)$ with no real points consists of three conjugate lines forming a triangle in the plane $PG(2, q^3)$ over a cubic extension of the base field $GF(q)$.

Proof. Let Π be a plane of $PG(5, q)$ not intersecting Ω , hence not intersecting any degenerate conic of $PG(2, q)$. In order to be able to make use of Dickson-Segre's Theorem, we embed the Desarguesian projective plane into $PG(2, q^3)$ and let σ be the automorphic period 3 collineation of $PG(2, q^3)$ that fixed the points of $PG(2, q)$. This implies that for any point P of $PG(2, q)$ the points P, P^σ and P^{σ^2} form a triangle. Furthermore, we denote by \mathcal{V}^* and Ω^* the extensions in $PG(5, q^3)$ of the Veronese surface \mathcal{V} and Ω , respectively.

Since Ω is a cubic hypersurface in $PG(5, q)$, by Dickson-Segre's theorem, Π intersects Ω^* in three conjugate lines, say l, l^σ and l^{σ^2} , forming a triangle. In order to identify these lines, recall that Ω^* corresponds to the degenerate conics in $PG(2, q^3)$. From the discussion in Section 4.1 we have that there is a one-to-one correspondence of pencils of degenerate conics in $PG(2, q^3)$ with the lines of $PG(5, q^3)$ completely contained in Ω^* . Indeed, there are only four types of pencils of degenerate conics in $PG(2, q^3)$ (see [11, Table 1.2.3], for detailed description) which implies follows that there are exactly the following four types of lines completely contained in Ω^* :

- i) Lines passing through two points of \mathcal{V}^* ,

- ii) Lines in a conic plane A of \mathcal{V}^* not intersecting \mathcal{V}^* ,
- iii) Lines being in the intersection of a tangent plane of \mathcal{V}^* at a point P of \mathcal{V}^* with a conic plane A through P ,
- iv) Lines of a tangent plane T of \mathcal{V}^* not passing through the point of tangency of \mathcal{V}^* .

From this characterization we can deduce that a line of $PG(2, q^3)$ is completely contained in Ω^* if and only if it belongs to either a conic plane of \mathcal{V}^* or a tangent plane of \mathcal{V}^* . Note that the first three types are the ones belonging to a conic plane of \mathcal{V}^* and the latter one belongs to a tangent plane of \mathcal{V}^* . Let us consider the two cases separately.

Assume first that the line l mentioned above belongs to a conic plane A of \mathcal{V}^* . Therefore the lines l^σ and l^{σ^2} belong to the conic planes A^σ and A^{σ^2} of \mathcal{V}^* , respectively. By Definition 4.3.2 item iii), we have that any two distinct conic planes of \mathcal{V}^* intersect in a unique point of \mathcal{V}^* . Since the lines l, l^σ and l^{σ^2} belong to three pairwise distinct conic planes, the vertices P, P^σ and P^{σ^2} of the triangle formed by these lines are points of \mathcal{V}^* which, by Definition 4.3.2 item ii), are points of a non-degenerate conic in $PG(2, q)$.

Secondly, assume that the line l belongs to a tangent plane T of \mathcal{V}^* and does not intersect \mathcal{V}^* . Then, similarly, the lines l^σ and l^{σ^2} correspond to the tangent planes T^σ and T^{σ^2} , respectively. From Definition 4.3.2 of a tangent plane and since the lines l, l^σ and l^{σ^2} all lie on pairwise distinct tangent planes it follows that they mutually intersect in a unique point. Explicitly this means

$$l \cap l^\sigma = P, l^\sigma \cap l^{\sigma^2} = P^\sigma \text{ and } l \cap l^{\sigma^2} = P^{\sigma^2},$$

where P, P^σ and P^{σ^2} are points of Ω^{e^*} . Therefore we have that P and P^σ belong to the same tangent plane T^σ and thus we can express the matrix coordinates of the three points by those of a line pair in $PG(2, q^3)$. Let, for instance, (a, b) represent the line pair in $PG(2, q^3)$ whose coordinates correspond to the matrix coordinates of P . Since P^σ belongs to the same tangent plane T^σ , its matrix coordinates can be represented by those of a line pair (b, c) in $PG(2, q^3)$. It follows that

$$(a, b)^\sigma = (a^\sigma, b^\sigma) = (b, c),$$

and $b = b^\sigma$, which implies that $b = a^\sigma$ and $c = b^\sigma = b^{\sigma^2}$.

Analogously we have that the matrix coordinates of P^{σ^2} correspond to the coordinates of the line pair (c, a) in $PG(2, q^3)$. Hence the matrix coordinates of P, P^σ and P^{σ^2} are those of the line pairs (a, a^σ) , (a^σ, a^{σ^2}) and (a^{σ^2}, a) , respectively.

We have seen before that P, P^σ and P^{σ^2} lie all on distinct conic planes. This implies that no point of the Desarguesian plane lies on the line a . Let us denote the intersection of two line pairs by

$$a \cap a^\sigma = A, a^\sigma \cap a^{\sigma^2} = A^\sigma \text{ and } a \cap a^{\sigma^2} = A^{\sigma^2}.$$

Note that the three intersection points are all points of $PG(2, q^3)$. Furthermore, let $\Pi = \langle P, P^\sigma, P^{\sigma^2} \rangle$. Then the matrix coordinates of all points of Π coincide with the

matrices of conics of $PG(2, q^3)$ passing through the points A, A^σ and A^{σ^2} . On the other hand, since a is a line of $PG(2, q^3)$, the plane Π of $PG(2, q^3)$ can be represented by the line pairs (a, a^σ) , (a^σ, a^{σ^2}) and (a^{σ^2}, a) described above and Π does not intersect Ω^* . Furthermore the plane $\Pi = \langle P, P^\sigma, P^{\sigma^2} \rangle$ is the set of all points of $PG(5, q^3)$ whose matrix coordinates correspond exactly to those of the conics of $PG(2, q^3)$ passing through A, A^σ and A^{σ^2} .

To summarize, we have now found two types of planes of $PG(5, q)$ which do not intersect in any degenerate conic of $PG(2, q)$ with the help of the characterization of a line. These two types are:

Type 1: A plane that intersects \mathcal{V}^* in the points P, P^σ and P^{σ^2} and that intersects Ω^* in the points of the three lines $\langle P, P^\sigma \rangle$, $\langle P^\sigma, P^{\sigma^2} \rangle$ and $\langle P, P^{\sigma^2} \rangle$, where these line belong to some conic plane of \mathcal{V}^* .

Type 2: A plane corresponding to the conics of $PG(2, q^3)$ passing through three points A, A^σ and A^{σ^2} , where the lines passing through two of them are not of $PG(2, q)$. This plane intersects Ω^* in the three lines belonging to a tangent plane of \mathcal{V}^* .

Finally, we need to determine which of the following types is a field plane and which one a semi-field plane. For this, let P and Q be distinct points of a line m of $PG(5, q)$ which does not intersect Ω .

Claim: Type 1 describes a field plane $\Pi = \langle P, Q, PQ^{-1}Q \rangle$.

Proof: The plane $\Pi = \langle P, Q, PQ^{-1}Q \rangle$ is indeed a field plane, by the choice of P and Q and hence, by Definition 4.4.1, PQ^{-1} satisfies a cubic equation which is irreducible over $GF(q)$. Say this cubic equation is given by

$$(PQ^{-1} - \lambda\mathbb{I})(PQ^{-1} - \lambda^q\mathbb{I})(PQ^{-1} - \lambda^{q^2}\mathbb{I}) = 0,$$

where $\lambda \in GF(q^3) \setminus GF(q)$. Hence the eigenvalues of PQ^{-1} are λ, λ^q and λ^{q^2} . We denote the corresponding eigenvectors by \mathbf{x}, \mathbf{x}^q and \mathbf{x}^{q^2} , respectively. Note that they are linearly independent over $GF(q^3)$. Furthermore, note that $(PQ^{-1} - \lambda\mathbb{I})$ and $(PQ^{-1} - \lambda^q\mathbb{I})$ commute. This leads to the equation

$$(PQ^{-1} - \lambda\mathbb{I})(PQ^{-1} - \lambda^q\mathbb{I})\mathbf{x} = (PQ^{-1} - \lambda\mathbb{I})(PQ^{-1} - \lambda^q\mathbb{I})\mathbf{x}^q = \mathbf{0},$$

where $\mathbf{0}$ denotes the all-zero vector. Hence, $(PQ^{-1} - \lambda\mathbb{I})(PQ^{-1} - \lambda^q\mathbb{I})$ is of rank 1 and thus $(PQ^{-1} - \lambda\mathbb{I})(PQ^{-1} - \lambda^q\mathbb{I})Q$ is of rank 1 as well. Since it is of rank 1, it corresponds to \mathcal{V}^* . But it belongs to Π as well. Therefore

$$(PQ^{-1} - \lambda\mathbb{I})(PQ^{-1} - \lambda^q\mathbb{I})Q \in \Pi \cap \mathcal{V}^*.$$

By the analogous arguments it holds that

$$\begin{aligned} (PQ^{-1} - \lambda^q\mathbb{I})(PQ^{-1} - \lambda^{q^2}\mathbb{I})Q &\in \Pi \cap \mathcal{V}^* \quad \text{and} \\ (PQ^{-1} - \lambda\mathbb{I})(PQ^{-1} - \lambda^{q^2}\mathbb{I})Q &\in \Pi \cap \mathcal{V}^*. \end{aligned}$$

This implies that the field plane $\Pi = \langle P, Q, PQ^{-1}Q \rangle$ is indeed of type 1 above. \blacksquare

If the plane $\Pi = \langle P, Q, PQ^{-1}Q \rangle$ is not a field plane then $(PQ^{-1} - \lambda\mathbb{I})$ is of rank 2. A symmetric, homogeneous (3×3) matrix of rank 2 represents either the points of Ω^{e^*} or the points of Ω^{i^*} . This implies that the line m passing through P and Q intersects in the three points $(PQ^{-1} - \lambda\mathbb{I})Q$, $(PQ^{-1} - \lambda^q\mathbb{I})Q$ and $(PQ^{-1} - \lambda^{q^2}\mathbb{I})Q$. Hence the plane Π does not intersect \mathcal{V}^* . Finally, we have that the conics of $PG(2, q)$ that are represented by the same matrices as the line m is a pencil of conics passing through a point of $PG(2, q)$ and the three conjugate points A , A^σ and A^{σ^2} of $PG(2, q^3) \setminus PG(2, q)$. Thus m belongs to a unique plane which we refer to a semi-field plane. This semi-field plane is therefore indeed of type 2 above. \square

With this theorem we finally have classified the projective bundles in $PG(2, q)$ by field-planes and semi-field planes of $PG(5, q)$. By Corollary 4.2.2 and Theorem 4.4.5, we are certain that projective bundles in $PG(2, q)$ exist. From the construction of the type 2 plane in the proof it follows as well that this semi-field plane is commutative. Indeed, the following corollary is an immediate consequence of the construction in the proof and Corollary 4.2.2.

Corollary 4.4.6. [11, Corollary 1.2.4] *There are essentially exactly two commutative semi-field planes of dimension three over $GF(q)$, where q is odd. These are first the field $GF(q^3)$ and second the semi-field plane of order q^3 of type 2 in the proof of the previous theorem.*

Additionally, we have seen that there is a one-to-one correspondence between the points of the Veronese surface in $PG(5, q)$ and the conics of $PG(2, q)$ whose associated matrices are of rank 1. With the classification of projective bundles of $PG(2, q)$, Theorem 4.4.5, and the characterization of field planes in $PG(5, q)$, Theorem 4.4.4, it is possible to prove the identification of a semi-field plane of $PG(5, q)$ with a Veronese surface \mathcal{V} . We will at this point only state the theorem without proving it. A detailed proof can be found in [11].

Theorem 4.4.7. [11, Theorem 1.2.7] *Let Π be a semi-field plane of $PG(5, q)$. Then the set of $q^2 + q + 1$ points $\Pi^{-1} = \{P^{-1} \mid P \in \Pi\}$ is isomorphic to a Veronese surface.*

Concluding this section we have identified three projective bundles of $PG(2, q)$. Baker et al. have named them in the following manner (see [2] Chapter 3):

1. *Circumscribed bundles:* These are the projective bundles of $PG(2, q)$ corresponding to the field $GF(q^3)$ in $PG(5, q)$. These projective bundles indeed exist for even order q as well. A circumscribed bundle in $PG(2, q)$ contains all the three vertices of the triangle $PP^\sigma P^{\sigma^2}$ introduced in the proof of Theorem 4.4.5.
2. *Inscribed bundles:* They are identified with the semi-field planes of $PG(5, q)$ and are hence projective bundles of $PG(2, q)$ corresponding to the points of a Veronese surface, by Theorem 4.4.7. These bundles consist of non-degenerate conics being tangent to the three sides of the triangle $PP^\sigma P^{\sigma^2}$ and they exist only for odd q .

3. *Self-polar bundles:* Finally this class of projective bundles of $PG(2, q)$ corresponds to a field plane of $PG(5, q)$ and consists of non-degenerate conics with respect to which the triangle $PP^\sigma P^{\sigma^2}$ is self-polar.

4.5 Alternative Representation

We have now classified all projective bundles by embedding $PG(2, q)$ into $P(2, q^3)$ and using the properties of conic and tangent planes of a Veronese surface \mathcal{V} in $PG(5, q)$. As mentioned above, Baker et al. have named the three classes of projective bundles. In their paper (see [2], Chapter 3) they present an alternative parametrization for each of the three projective bundles, which is of a more algebraic nature and more intuitive to apply. In this section we will study this representation.

Still consider the projective plane $PG(2, q)$, where q is an odd prime power. The key idea, compared to Glynn's idea, is not going to five dimensions but instead to identify each point of the Desarguesian plane $PG(2, q)$ with an integer modulo $N = q^2 + q + 1$.

We would like to have a general description for the set of lines. Let us therefore introduce perfect difference sets.

Definition 4.5.1. [15, page 177] If a set A of $r + 1$ distinct integers a_0, \dots, a_r has the property that $(a_i - a_j)_{0 \leq i < j \leq r}$ are distinct mod $r^2 + r + 1$, then A is called a *perfect difference set*.

Example 4.5.2. Let $r = 2$ then the set $A = \{1, 2, 4\}$ is a perfect difference set, since the differences

$$2 - 1 = 1, \quad 1 - 2 = -1, \quad 4 - 2 = 2, \quad 2 - 4 = -2, \quad 4 - 1 = 3 \quad \text{and} \quad 1 - 4 = -3$$

are all distinct mod $2^2 + 2 + 1 = 7$.

However, perfect difference sets do not exist for any choice of r . Similar to the existence of finite fields, Singer has shown in his paper [34] from 1938 that perfect difference sets exist, whenever r is a prime power. In our setting the number r is the order q of the projective plane $PG(2, q)$ which, by assumption, is an odd prime power. Therefore a perfect difference set can exist in our case.

Perfect difference sets are used to describe a line in $PG(2, q)$. But since there are various perfect difference sets, a natural question arising is: How to construct a perfect difference set? We will follow the instructions given by Hirschfeld (see [17] pages 77 - 79 for further details).

Let $A_0, A_1, \dots, A_{q^2+q+1}$ be the points of $PG(2, q)$. Each of these points A_i has a coordinate representation $A_i = (a_i, b_i, c_i)$, where $a_i, b_i, c_i \in GF(q)$ not all zero. Let λ be a generator element of the multiplicative group of $GF(q^3)$. This means that $\{1, \lambda, \lambda^2\}$ forms a basis for $GF(q^3)$. Hence, every element of $GF(q^3)$ (and thus each power of λ) can be described by λ^2 and λ using the coordinates of the points A_i of $PG(2, q)$, i.e.

$$\lambda^i = a_i \cdot \lambda^2 + b_i \cdot \lambda + c_i.$$

Conversely, of course the three elements a_i , b_i and c_i , for $i = 0, \dots, q^2 + q$, can uniquely be determined by a power of λ and hence they are all non-zero elements of $GF(q^3)$.

Remember that we identify the points of $PG(2, q)$ with the integers modulo $q^2 + q + 1$. Let us focus on the points A_0 , which will be the point of $PG(2, q)$ identified with 0, and A_1 , which analogously will be the point of $PG(2, q)$ identified with the integer 1. We are looking for points of $PG(2, q)$ which are collinear to A_0 and A_1 . Namely, point A_i of $PG(2, q)$ different from A_0 and A_1 is collinear to A_0 and A_1 if there exist non-zero elements $k_0, k_1, k_2 \in GF(q)$ such that

$$k_0 \cdot \lambda^0 + k_1 \cdot \lambda + k_2 \cdot \lambda^i = 0,$$

which can be simplified to

$$1 + k \cdot \lambda + l \cdot \lambda^i = 0,$$

for k and l some non-zero elements of $GF(q)$. Since we identify the points of $PG(2, q)$ with the integers modulo $q^2 + q + 1$, a point A_i is identified with the integer i for $i = 0, \dots, q^2 + q + 1$.

We will rename A_0 and A_1 by d_0 and d_1 , respectively. Suppose that the points of $PG(2, q)$ collinear with d_0 and d_1 are labelled by d_2, \dots, d_q . Consider then the following array

$$\mathcal{A} = \begin{pmatrix} d_0 & d_0 + 1 & \dots & d_0 + q^2 + q \\ d_1 & d_1 + 1 & \dots & d_1 + q^2 + q \\ \vdots & \vdots & \dots & \vdots \\ d_q & d_q + 1 & \dots & d_q + q^2 + q \end{pmatrix}.$$

Then the following theorem by Hirschfeld holds.

Theorem 4.5.3. [17, Theorem 4.2.2 and Corollary] *The matrix \mathcal{A} described above with integers reduced mod $q^2 + q + 1$ represents the points and lines of $PG(2, q)$. Moreover, the set $D = \{d_0, \dots, d_q\}$ is a perfect difference set.*

Proof. Observe, that the first row of \mathcal{A} (when reduced modulo $q^2 + q + 1$) represents all the points of $PG(2, q)$, since the points of $PG(2, q)$ are identified with the integers modulo $q^2 + q + 1$ and d_0 is referred to the point identified with 0. Since we chose d_2, \dots, d_q to be the points which are collinear with the points identified with d_0 and d_1 , the first column represents a line passing through these $q + 1$ points. As a consequence, each successive column is a shifting of the previous one by 1. Hence, each successive column also represents the points of a line. Therefore the matrix \mathcal{A} with integers reduced modulo $q^2 + q + 1$ represents the points and lines of $PG(2, q)$. In order to show, that D is a perfect difference set, let us consider the matrix \mathcal{A}' formed by the columns of \mathcal{A} which contain the point denoted by d_0 . If we reduce then modulo $q^2 + q + 1$, we obtain

$$\mathcal{A}' = \begin{pmatrix} d_0 - d_0 & d_0 - d_1 & \dots & d_0 - d_q \\ d_1 - d_0 & d_1 - d_1 & \dots & d_1 - d_q \\ \vdots & \vdots & \dots & \vdots \\ d_q - d_0 & d_q - d_1 & \dots & d_q - d_q \end{pmatrix}.$$

Hence, each column represents a line passing through the point denoted by d_0 . Apart from the point denoted by d_0 each point lies just on one of these lines. Thus, apart from the main diagonal, the $q^2 + q$ integers in the matrix \mathcal{A}' are distinct and consist of all the differences $d_i - d_j$. If we apply the same idea to each point of the matrix \mathcal{A} , we can deduce that every column of \mathcal{A} forms a perfect difference set. Hence, the result follows. \square

We have implemented the computation of the perfect difference sets in Python using SageMath (see Listing 1 in Appendix A). The following table shows the perfect difference sets for some values of q .

q	perfect difference set D
2	$\{0, 1, 3\}$
3	$\{0, 1, 3, 9\}$
5	$\{0, 1, 3, 8, 12, 18\}$
7	$\{0, 1, 3, 13, 32, 36, 43, 52\}$
9	$\{0, 1, 3, 9, 27, 49, 56, 61, 77, 81\}$

Table 4.1: Perfect difference sets for some initial values of q .

The above theorem yields that the lines of $PG(2, q)$ are the images of the perfect difference set D under the Singer cycle $S(i) = i + 1$ for all integers $i \in \mathbb{Z}/\langle q^2 + q + 1 \rangle$. Hence the set of lines is given by

$$\mathcal{L} = \{ \{d_0 + i, d_1 + i, \dots, d_q + i\} \mid i \in \mathbb{Z}/\langle q^2 + q + 1 \rangle \}.$$

For instance, consider Example 3.3.4 given in Section 3.3, the Fano plane $PG(2, 2)$. Instead of labelling the points by their coordinates, we can identify each point with an integer modulo 7. Hence the set of points is

$$\mathcal{P} = \{0, 1, 2, 3, 4, 5, 6\}.$$

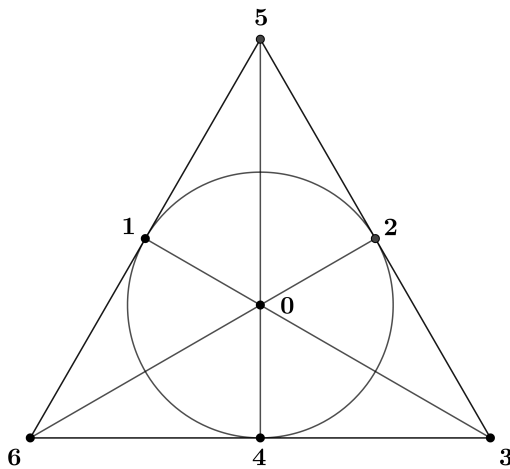


Figure 4.2: The Fano plane: Points identified with integers mod 7.

From the figure we can see that the set of lines is given by

$$\mathcal{L} = \{\langle 0, 1, 3 \rangle, \langle 0, 2, 6 \rangle, \langle 0, 4, 5 \rangle, \langle 1, 5, 6 \rangle, \langle 1, 2, 4 \rangle, \langle 2, 3, 5 \rangle, \langle 3, 4, 6 \rangle\}.$$

Since we will mainly focus on the case, where q is an odd prime power we give another example for $q = 3$.

Example 4.5.4. Consider the projective plane $PG(2, 3)$. The order of the plane is $q = 3$. Then the set of points is given by the integers mod $q^2 + q + 1 = 13$, i.e. $\mathcal{P} = \{0, 1, \dots, 12\}$. From table 4.1 we obtain that the perfect difference set is $D = \{0, 1, 3, 9\}$ which implies that the set of lines is given by

$$\mathcal{L} = \{\{0 + i, 1 + i, 3 + i, 9 + i\} \mid i \in \mathbb{Z}/\langle q^2 + q + 1 \rangle\}.$$

The incidence matrix of points and lines of $PG(2, q)$ is then defined analogously to the definition given in Section 3.1. Hence we obtain an incidence matrix of the form

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

Having the tools ready we are able to identify the projective bundles. As we have discussed in Section 4.1, projective bundles can be viewed as lines in $PG(2, q)$. Thus we can imagine that a projective bundle, which is a set of non-degenerate conics, of $PG(2, q)$ has a similar description as the set of lines in $PG(2, q)$. Indeed the following theorem holds.

Theorem 4.5.5. [2, Theorem 3.2] *For $N = q^2 + q + 1$, if $r \in \mathbb{Z}/\langle N \rangle$ is relatively prime to N and $D = \{d_0, \dots, d_q\}$ is a perfect difference set for $PG(2, q)$, then the set $D/r = \{d_0/r, \dots, d_q/r\}$ is the point set of some curve of degree r .*

Note that the elements d_i/r are computed mod N for each $i \in \mathbb{Z}/\langle N \rangle$.

In our setting, q is an odd prime power and $N = q^2 + q + 1$ is odd too. Hence the values $r \in \{-1, 2^{-1}, 2\}$ are always relatively prime to N . As a consequence, we get with the help of Bruck (see [7, Chapter 8]) and Hall (concern [16, Theorem 4.5 and p.1085]) that the three projective bundles classified by Glynn can be represented in the following way:

1. *Circumscribed bundle:* Can be described as the image of $D/(-1) = -D$ under the Singer cycle $S(i) = i + 1$, i.e.

$$\mathcal{B}_C = \{\{-d_0 + i, -d_1 + i, \dots, -d_q + i\} \mid i \in \mathbb{Z}/\langle q^2 + q + 1 \rangle\}.$$

A conic of a circumscribed bundle satisfies the equation $x^{-1} + y^{-1} + z^{-1} = 0$ or equivalently $yz + xz + xy = 0$.

2. *Inscribed bundle:* An inscribed bundle is characterized by the image of $D/(1/2) = 2D$ under the Singer cycle, which means

$$\mathcal{B}_I = \{\{2 \cdot d_0 + i, 2 \cdot d_1 + i, \dots, 2 \cdot d_q + i\} \mid i \in \mathbb{Z}/\langle q^2 + q + 1 \rangle\}.$$

This implies that it satisfies an equation $x^{1/2} + y^{1/2} + z^{1/2} = 0$ or equivalently $x^2 + y^2 + z^2 - 2xy - 2xz - 2yz = 0$.

3. *Self-polar bundle:* The set $D/2$ describes a non-degenerate conic for which the triangle with vertices P , P^σ and P^{σ^2} from Section 4.4 is a self-polar triangle. Hence S takes $D/2$ to the self-polar bundle, namely

$$\mathcal{B}_S = \{\{d_0 \cdot 2^{-1} + i, d_1 \cdot 2^{-1} + i, \dots, d_q \cdot 2^{-1} + i\} \mid i \in \mathbb{Z}/\langle q^2 + q + 1 \rangle\}.$$

To get a more intuitive understanding let us give a short example for $q = 3$.

Example 4.5.6. Consider the finite projective plane $PG(2, 3)$. As we have already discussed in Example 4.5.4, a perfect difference set is given by $D = \{0, 1, 3, 9\}$. Therefore the three types of projective bundles are given by the following sets:

1. The circumscribed bundle is described by the shifts of $-D$, which in our case will be the set

$$-D = \{0, -1, -3, -9\} = \{0, 4, 10, 12\}.$$

Hence a circumscribed bundle in $PG(2, 3)$ is given by

$$\begin{aligned} \mathcal{B}_C &= \{\{0 + i, 4 + i, 10 + i, 12 + i\} \mid i \in \mathbb{Z}/\langle q^2 + q + 1 \rangle\} \\ &= \{\{0, 4, 10, 12\}, \{1, 5, 11, 0\}, \{2, 6, 12, 1\}, \{3, 7, 0, 2\}, \{4, 8, 1, 3\}, \\ &\quad \{5, 9, 2, 4\}, \{6, 10, 3, 5\}, \{7, 11, 4, 6\}, \{8, 12, 5, 7\}, \{9, 0, 6, 8\}, \\ &\quad \{10, 1, 7, 9\}, \{11, 2, 8, 10\}, \{12, 3, 9, 11\}\}. \end{aligned}$$

2. For the inscribed bundle we first compute $2D$. Hence we obtain

$$2D = \{0, 2, 6, 13\} = \{0, 2, 5, 6\}.$$

Then an inscribed bundle in $PG(2, 3)$ corresponds to

$$\mathcal{B}_I = \{\{0 + i, 2 + i, 5 + i, 6 + i\} \mid i \in \mathbb{Z}/\langle q^2 + q + 1 \rangle\}.$$

3. Lastly we compute $D/2$, where $2^{-1} = 7 \pmod{q^2 + q + 1 = 13}$. This implies that

$$D/2 = 7D = \{0, 7, 21, 63\} = \{0, 7, 8, 11\}.$$

Hence a self-polar bundle in $PG(2, 3)$ will be given by the set

$$\mathcal{B}_S = \{\{0 + i, 7 + i, 8 + i, 11 + i\} \mid i \in \mathbb{Z}/\langle q^2 + q + 1 \rangle\}.$$

Chapter 5

Moderate Density Parity-Check Codes

In this chapter Moderate Density Parity-Check Codes (MDPC-Codes) are introduced. We will first give a little background on why they are important in cryptography, then give a few necessary definitions. Definitions and notions have been taken from various sources such as Gallager's PhD Thesis on Low-Density Parity-Check Codes (LDPC-Codes) [10] or Tillich's (joint) papers on MDPC-Codes ([37] and [24]).

5.1 Motivation and Definitions

In 1963 Gallager developed the *Low Density Parity-Check Codes* ([10]) which are binary linear error-correcting codes with a sparse parity-check matrix and constant row- and column-weight. This means that the matrix has just a few number of one-entries in each row and column and that this number of one-entries is the same for each row and is the same for each column. This class of error-correcting codes have been forgotten for a long time but were restudied and came up as suggestions for the McEliece cryptosystems in various papers ([25], [3]). LDPC-codes were often generated randomly by a computer and with the help of belief propagation processes the generator was able to produce codes providing a very good performance (actually extremely close to Shannon limit). Randomly generated LDPC-codes have no algebraic structure and small minimum distance. Due to their sparse parity-check matrix they provide an efficient and fast error-correction performance. This made them seem to be good candidates for some McEliece cryptosystem variants. LDPC-codes were studied more and more and there have been presented various combinatorial and algebraic constructions (see for example [22] or [38]). These codes then have an algebraic structure and are therefore easier to encode. Even their description is simpler to understand and the minimum distance is better. Later more and more LDPC-codes were constructed using finite geometry (such as [21], [39]). Nevertheless, there is one serious disadvantages in using LDPC-codes in the McEliece cryptosystem: insecurity. The parity-check matrix is of low weight and since a parity-check matrix generates the dual code, the rows of the parity-check matrix can be identified with codewords of the dual code, which then are also of low weight. This makes the use of LDPC-codes in the McEliece cryptosystem inse-

cure since an attack consists of finding dual low weight codewords and building a sparse parity-check matrix with them. Hence one can reconstruct a part of the code.

The idea was then to slightly increase the row-weight of the parity-check matrix. This is how *moderate density parity-check codes* got introduced. Even though all known attacks seem to be avoided, the error-correction performance becomes significantly less efficient in comparison with LDPC-codes, but it is sufficient to provide efficient decoding algorithms. In coding theory the focus mainly lies on correcting a certain number of errors to ensure the trustworthiness of a communication and not in correcting as many errors as possible. MDPC-codes fulfill this requirement and are thus adequate candidates for a cryptosystem.

Throughout the whole paper we define an MDPC-code in the following way.

Definition 5.1.1. [37, Definition 1] A *moderate density parity-check code*, or simply MDPC-code, is a binary linear code of length n with a parity-check matrix H whose row weight is $\mathcal{O}(\sqrt{n})$. If the weight of every column of H is a constant v and the weight of every row of the H is a constant w we say the MDPC-code is *of type* (v, w) .

Hence MDPC-codes are linear block codes which use a parity-check matrix in a decoder. Our goal is to be capable to correct as many errors as possible in one round of a decoding algorithm. To be able to study the amount of errors that can be corrected, another quantity will be needed which is the maximum column intersection introduced in the following definition.

Definition 5.1.2. [37, Definition 2] Let $H = (h_{ij})_{\substack{1 \leq i \leq r \\ 1 \leq j \leq n}}$ be a binary matrix. The *intersection number* of two different columns j and j' of H is equal to the number of rows i for which $h_{ij} = h_{ij'} = 1$. The *maximum column intersection* of H , denoted s_H , is equal to the maximum intersection number of two distinct columns of H .

Example 5.1.3. Consider a $[7, 4]$ -binary linear code with parity-check matrix

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

We can see that any two different columns have an intersection number of either zero, one or two. Therefore H has maximum column intersection $s_H = 2$.

The notion of maximum column intersection will be needed to analyse and understand the error-correction capability and decoding performance of MDPC-codes. We will also see, that a small maximum column intersection provides a better error-correction capability.

5.2 Decoding Algorithms

There exist several decoding algorithms for LDPC and MDPC-codes such as Gallager A, Gallager B or the bit-flipping decoding algorithm (see [10] and [13]). In

his Ph.D. thesis, Gallager has presented these decoding algorithms for LDPC-codes. Janoska has modified Gallager bit-flipping algorithm for MDPC-codes (see [18] p. 1087). We will introduce some well-known decoding algorithms in the section here. Later, we will only focus on the bit-flipping decoding algorithm and on the modified version presented by Artur Janoska and analyse its performance using a parity-check matrix of MDPC-codes.

5.2.1 The Bit-Flipping Algorithm

Let us start with the bit-flipping decoding algorithm, introduced by Gallager in 1963 for LDPC-codes. As the name of the algorithm might already suggest, in the bit-flipping decoding algorithm some bits of the received word will be flipped in order to get back the original message, whereas a *bit* refers to an entry of a binary linear code. This decoding algorithm is well-known and simple to use. Unfortunately, it only works over a binary alphabet, i.e. over $GF(2)$. A modified version of the bit-flipping decoding algorithm for MDPC-codes was later presented in [24].

The bit-flipping decoding algorithm works as follows. Assume that a parity-check matrix H of size $(r \times n)$ for a binary linear code C is given and say that a word y is received. The decoding algorithm computes for each column j in a parity-check matrix H the number of one-entries n_j and compares it to the number of unsatisfied check equations u_j involving bit j , i.e.

$$u_j = |\{i \in \{1, \dots, r\} \mid h_{ij} = 1, \sum_l h_{il}y_l = 1 \pmod{2}\}|.$$

If for a certain bit j the number of unsatisfied check equation is more than half of the number of one-entries in the j -th position of the received word y , the bit in the corresponding column gets flipped (i.e. a one-entry becomes a zero-entry and vice-versa) and the syndrome is recomputed. The algorithm stops if either the syndrome becomes zero or if a maximum number of iterations b_{max} is reached.

The decoding algorithm has complexity $\mathcal{O}(nwb)$, where n is the code-length, w the column weight and b the average number of iterations. Furthermore, the error-correction capability is depending on two sizes of the code; its length and its row-weight. Indeed it is linearly increasing with the code-length and almost linearly decreasing with the row-weight. Since the row-weight of MDPC-codes is increased when compared to LDPC-codes, the error-correction capability decreases.

Rafael Misoczki et. al. have suggested three modifications for choosing the number of iterations b in order to minimize the problem (see [24]):

1. Precompute a sequence of b 's (see [10], p. 46, Inequality (4.16)).
2. At each iteration, choose b to be the maximum number of unsatisfied parity-check equations, denoted by Max_{upc} .
3. For a variable small integer δ let $b := \text{Max}_{upc} - \delta$.

The following shows the pseudocode of the bit-flipping decoding algorithm for MDPC-codes([18]).

Algorithm 1 Bit-flipping decoding algorithm

Require: Parity-check matrix $H = (h_{i,j}) \in \{0, 1\}^{r \times n}$,
 received codeword $y \in \{0, 1\}^n$,
 maximal number of iterations b_{max}

Ensure: decoded codeword.

```

 $s \leftarrow Hy^\top$  ▷ compute the syndrome.
for  $j = 1$  to  $n$  do
   $n_j \leftarrow |\{i \in \{1, \dots, r\} \mid h_{ij} = 1\}|$  ▷ Number of 1-entries per column.
end for

for  $a = 1$  to  $b_{max}$  do ▷ Round 1 to  $b_{max}$  of algorithm.
  for  $j = 1$  to  $n$  do
     $u_j \leftarrow |\{i \in \{1, \dots, r\} \mid h_{ij} = 1, \sum_l h_{il}y_l = 1 \pmod{2}\}|$  ▷ Number of
    unsatisfied parity checks.
  end for
  for  $j = 1$  to  $n$  do
    if  $u_j > n_j/2$  then ▷ Flipping condition.
       $y_j \leftarrow 1 - y_j$  ▷ Flip bit  $j$ .
       $s = Hy^\top$  ▷ Recompute the syndrome.
    end if
  end for
  if  $s = 0$  then ▷ Algorithm stops if syndrome is the zero-vector.
    return  $y$ 
  end if
end for
return error
  
```

Example 5.2.1. Consider a binary linear code C with parity-check matrix

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

and assume that the received word is $y = (1, 0, 1, 0, 1, 1)$.

Step 1: Compute the syndrome, $s = Hy^\top = (0, 1, 0, 0)^\top$. So we obtain, that the syndrome is not the all-zero vector, which means that there is an error in the received word.

Step 2: Compute for each column j the number of nonzero entries n_j . We obtain

$$n_j = 2, \text{ for all } j = 1, \dots, 6.$$

Step 3: For each column we compute the number of unsatisfied parity-check equations $u_j = |\{i \in \{1, \dots, r\} \mid h_{ij} = 1, \sum_l h_{il}y_l = 1 \pmod{2}\}|$.

$$\begin{aligned} u_1 &= |\{i = 2, 4\}| = 2, \\ u_2 &= |\{i = 4\}| = 1, \\ u_3 &= |\{\}| = 0, \\ u_4 &= |\{i = 4\}| = 1, \\ u_5 &= |\{i = 2\}| = 1, \\ u_6 &= |\{i = 2\}| = 1. \end{aligned}$$

Step 4: This is the flipping condition. Check now for each column j if $u_j > n_j/2$. If so, then flip the bit y_j . We see that only for $j = 1$ the flipping condition is satisfied. Hence we flip the bit y_1 from 1 to 0. Therefore the updated received word is $y = (0, 0, 1, 0, 1, 1)$.

Step 5: Recompute the syndrome s . We now get that $s = Hy^\top = (0, 0, 0, 0)^\top$ is the allzero vector. Hence we have found the correct codeword $y = (0, 0, 1, 0, 1, 1)$. If the syndrome s was not the all-zero vector here, we would need to repeat steps 2 to 5.

Gallager has presented in [10] a computation of the threshold for the number of errors that can be corrected by an LDPC-code. This technique has been used in [24] for MDPC-codes. Even if it is not precise for MDPC-codes it provides an upper bound. In order to estimate the threshold of the bit-flipping algorithm we estimate the probability that a bit is in error after a given number of iterations. We then achieve a reliable error-correction if this probability converges to zero.

For this, consider an MDPC-code of length n and parity-check matrix H of size $(\frac{n}{2} \times n)$. We denote by $d_c = \mathcal{O}(\sqrt{n})$ the degree of the check nodes (i.e. the row-weight) and by $d_v = \mathcal{O}(\frac{\sqrt{n}}{2})$ the degree of the variable nodes (i.e. the column-weight). Furthermore, let p_0 denote the probability that a bit is received in error. With this we get

$$\begin{aligned} p_{i+1} = p_0 - p_0 \sum_{l=b_d}^{d_v-1} \binom{d_v-1}{l} \left(\frac{1 + (1-2p_i)^{d_c-1}}{2} \right)^l \left(\frac{1 - (1-2p_i)^{d_c-1}}{2} \right)^{d_v-l-1} \\ + (1-p_0) \sum_{l=b_d}^{d_v-1} \binom{d_v-1}{l} \left(\frac{1 + (1-2p_i)^{d_c-1}}{2} \right)^l \left(\frac{1 - (1-2p_i)^{d_c-1}}{2} \right)^{d_v-l-1}, \end{aligned}$$

where the integer b_d is chosen to be an integer between $d_v - 1$ and $d_v/2$ (see [24]) satisfying

$$\frac{1-p_0}{p_0} \leq \left(\frac{1 + (1-2p_i)^{d_c-1}}{1 - (1-2p_i)^{d_c-1}} \right)^{2b_d-d_v+1},$$

which minimizes the function p_i .

5.2.2 The Gallager A Algorithm

Another algorithm also due to Gallager is the Gallager A algorithm. In comparison to the bit-flipping algorithm, where we worked over the alphabet $GF(2)$, Gallager A uses the set $\mathcal{M} = \{-1, 1\}$ as alphabet. This algorithm also uses the alphabet of the channel, which will be denoted by \mathcal{Y} . There is also a Gallager B algorithm which extends the first one to an arbitrary long alphabet (see [10] or [14] for more information).

Gallager has defined this algorithm over a binary symmetric channel (BSC) with crossover probability p , which is the probability that a bit which was transmitted correctly is flipped. This means, if X is the transmitted variable and Y the received one, we get the following probabilities.

$$\begin{aligned}\mathbb{P}(Y = -1|X = -1) &= \mathbb{P}(Y = 1|X = 1) = p, \\ \mathbb{P}(Y = -1|X = 1) &= \mathbb{P}(Y = 1|X = -1) = 1 - p.\end{aligned}$$

Gallager's A algorithm is a message-passing iterative decoding algorithm. This means that messages are exchanged between variable nodes v_i and check nodes c_j in discrete time steps. Each check nodes processes the message received from its neighbours and sends back a suitable message in the alphabet \mathcal{M} to each of its neighbours. Then each variable uses that information together with its own received value r_i to produce new messages which are sent back to the neighbouring check nodes.

In order to define this message-passing process we need to define for each iteration step l the two message maps

$$\begin{aligned}\psi_v^{(l)} : \mathcal{Y} \times \mathcal{M}^{d_v-1} &\longrightarrow \mathcal{M}, \\ \psi_c^{(l)} : \mathcal{M}^{d_c-1} &\longrightarrow \mathcal{M},\end{aligned}$$

where $\psi_v^{(l)}$ is the message map for a variable node v of degree d_v and $\psi_c^{(l)}$ is the message map for a check node c of degree d_c . Then the algorithm works as follows.

Initially the algorithm says that its channel output is either -1 or 1 and it is going to request the variable to send out whatever it receives from the channel along all its outgoing edges, i.e.

$$\psi_v^{(0)}(m) = m,$$

where m is the message that is received.

Thereafter the variable node v is going to send out the input unless there is overwriting evidence provided by all of the other inputs. Meaning if all other inputs have a sign that disagrees with the sign of the channel input, then v sends the reverse sign,

$$\psi_v^{(l)}(m_0, m_1, \dots, m_{d_v-1}) = \begin{cases} -m_0, & \text{if } m_j = -m_0 \text{ for all } 1 \leq j \leq d_v - 1, \\ m_0, & \text{else.} \end{cases}$$

Finally the check node c sends the product of all incoming messages to v ,

$$\psi_c^{(l)}(m_1, \dots, m_{d_c-1}) = \prod_{j=1}^{d_c-1} m_j.$$

The computation of the threshold is the same as for the bit-flipping algorithm. The only difference is that the value b_i is concretely given by $b_i = \lfloor \frac{d_v+1}{2} \rfloor$, for all i .

We have computed some thresholds for Gallagers A algorithms using codes of lengths $n = 2^k$ for $k = 4, \dots, 10$. The following table shows the corresponding threshold for these code-lengths.

length n	d_v	d_c	threshold p_0
2^4	2	4	$8.8818 \cdot 10^{-17}$
2^5	2	5	$8.8818 \cdot 10^{-17}$
2^6	4	8	0.04762
2^7	5	11	0.025
2^8	8	16	0.00952
2^9	11	22	0.00476
2^{10}	16	32	0.00215

Table 5.1: Some threshold computations for MDPC-codes.

5.2.3 Belief Propagation

Belief propagation is another type of message-passing algorithm. It is sometimes also called sum-product algorithm. The algorithm, which was first introduced by Judea Pearl in 1982 (see [26]), computes for each variable node its marginal distribution, conditioned on any check node.

Compared with the other algorithms each variable and check node has a probabilistic value instead of a fixed value 0 or 1. It further has shown empirical success in the use with LDPC-codes.

Due to this fact, an interesting topic would be to study the performance of this algorithm on MDPC-codes, but we will not treat this in this master thesis.

5.3 Error-Correction Capacity

Now that we have the most important tools, we can study the error correction capability of MDPC-codes using the bit-flipping algorithm. With the maximum column intersection we can lower bound the worst-case error-correction performance for one round of bit-flipping and we can correct more errors for small maximum column intersection. In fact, Tillich has shown the following result.

Proposition 5.3.1. [37, Proposition 1] *Let C be an MDPC-Code of type (v, w) with parity-check matrix $H = (h_{ij})_{\substack{1 \leq i \leq r \\ 1 \leq j \leq n}}$. Let s be the maximum column intersection with respect to the parity check matrix H . Performing one round of the bit-flipping decoding algorithm based on the matrix H one can correct all errors of weight at most $\lfloor \frac{v}{2s} \rfloor$.*

Proof. Let $e = (e_1, \dots, e_n)$ denote the error vector and assume that t is the number of errors and that $t \leq \lfloor \frac{v}{2s} \rfloor$. We define the set of positions j in error that are in the support of the i -th parity-check equation by

$$E_i = \{j \in \{1, \dots, n\} \mid h_{ij} = 1 \text{ and } h_{ij}e_j = 1\}.$$

Now let us perform one round of bit-flipping and see what happens to y_j . There are two options: Either y_j is in error or not. Just as in the description of the algorithm we denote by u_j the number of unsatisfied parity-check equations involving bit j , similarly s_j is the number of satisfied parity-check equations and n_j the number of one-entries involving bit j .

1. **y_j is in error:** Use a bipartite graph G_j associated to bit j . Let A_j be the set of positions different from j being in error and B_j the set of parity-check equations involving bit j such that $|E_i| \geq 2$. Then define the set of vertices to be $A_j \cap B_j$. Therefore we have an edge between a position $l \in A_j$ and a parity-check equation $i \in B_j$ if and only if $h_{il} = 1$.

We note, that s_j can be upper bounded by the number of parity-check equations involving bit j satisfying $|E_i| \geq 2$.

$$\begin{aligned} s_j &\leq |\{i \in \{1, \dots, r\} \mid h_{ij} = 1 \text{ and } |E_i| \geq 2\}| \\ &\leq \mathcal{E}_j, \end{aligned}$$

where \mathcal{E}_j is the total number of edges of G_j .

Since s is the maximum column intersection, the degree of any vertex of G_j is smaller than s . Hence it follows

$$\begin{aligned} \mathcal{E}_j &\leq s \cdot |A_j| \\ &\leq s(t-1) \\ &\leq s \left(\left\lfloor \frac{v}{2s} \right\rfloor - 1 \right) \\ &\leq \frac{v}{2}, \end{aligned}$$

because A_j is the set of positions different from j that are in error and $t \leq \lfloor \frac{v}{2s} \rfloor$ by assumption. Finally since $\frac{v}{2} \leq \frac{n_j}{2}$, we conclude $s_j \leq \frac{n_j}{2}$ and the flipping condition is satisfied and y_j is flipped.

2. **y_j is correct:** Again we consider a bipartite graph G'_j consisting of the set of edges \mathcal{E}'_j and the set of vertices $A'_j \cap B'_j$, where A'_j is defined in the same way as A_j in the case before and let B'_j be the set of parity-check equations involving bit j satisfying $|E_i| \geq 1$. Then similarly there is an edge between a position $l \in A'_j$ and a parity-check equation $i \in B'_j$ if and only if $h_{il} = 1$.

Instead of upper-bounding s_j , we can upper bound u_j using the same argu-

ments as in the first case by

$$\begin{aligned}
u_j &\leq |\{i \in \{1, \dots, r\} \mid h_{ij} = 1 \text{ and } |E_i| \geq 1\}| \\
&\leq \mathcal{E}'_j \\
&\leq s \cdot |A'_j| \\
&\leq st \\
&\leq s \left(\left\lfloor \frac{v}{2s} \right\rfloor \right) \\
&\leq \frac{v}{2}.
\end{aligned}$$

Again since $\frac{v}{2} \leq \frac{n_j}{2}$ we obtain $u_j \leq \frac{n_j}{2}$. This means that the flipping condition is not satisfied and hence y_j is not flipped. □

We deduce this result: the smaller the maximum column intersection of a parity-check matrix of an MDPC-code, the more errors can be corrected within one round of the bit-flipping decoding algorithm. Hence a small column intersection effects good error-correction capability. In fact, if we use the random construction provided by Tillich (see [37], p. 5) we can construct MDPC-codes that are able to correct all errors of order $\mathcal{O}\left(\frac{\sqrt{n \cdot \log \log(n)}}{\log(n)}\right)$. More precisely with this construction for any $\epsilon > 0$ the maximum column intersection, with some probability $1 - o(1)$, is smaller than $(2 + \epsilon) \frac{\log(n)}{\log \log(n)}$.

The goal of this master thesis is to construct MPDC-codes of maximum column intersection equal to 2, which is formalized in Chapter 6.

5.4 McEliece Cryptosystem

As mentioned MDPC-codes were used in McEliece cryptosystem instead of Goppa codes, since Goppa codes had a too large key size. There exist different variants of this cryptosystem also depending on the different constructions of the code. Rafael Misoczki et al. have presented a variant using either a normal MDPC-code or a quasi-cyclic MDPC-code (see [24] for the definition and construction).

Their McEliece variant then works as follows.

1. Key-Generation:

Construct an MDPC-code of length n and row weight w that can correct up to t errors. For this, generate a parity-check matrix $H \in GF(2)^{r \times n}$ and its corresponding generator matrix with one of the possible constructions.

Public key: generator matrix G .

Private key: parity-check matrix H .

2. Encryption:

Let $m \in GF(2)^{n-r}$ be a message to be encrypted into $x \in GF(2)^n$. For this

generate a codeword $e \in GF(2)^n$ of weight less than t , $wt(e) \leq t$. Then the message m is encrypted by

$$x = mG + e.$$

3. Decryption:

For decryption choose a decoding algorithm ψ_H (such as the modified bit-flipping decoding algorithm) which knows the parity-check matrix H . To decrypt the received word $x \in GF(2)^n$ into the original message m , compute first

$$mG = \psi_H(x).$$

Then extract m from the first $(n - r)$ positions of mG .

Chapter 6

Construction

Finally, we are getting to the core of this master thesis. Namely, in this chapter we are going to present a new construction for MDPC-codes using finite geometry in the Desarguesian plane $PG(2, q)$ of odd order q . We make use of projective bundles, which exist in the Desarguesian plane as we have seen in Chapter 4, and the properties of the Desarguesian plane $PG(2, q)$ for a fixed odd prime power q . There have been several constructions for LDPC-codes based on geometry. For example Liu and Pados have constructed LDPC-codes from generalized polygons (for more information see [21]). Furthermore, there exist constructions of binary linear codes using conics in $PG(2, q)$ [33]. But the idea of using similar tools for MDPC-code as well is new.

MDPC-codes have been constructed in many different ways. One of the most frequent is to construct MDPC-codes as quasi-cyclic codes⁴ (see [24] and [18]). In that case a parity-check matrix is constructed from n_0 circulant matrices H_i all of size $p \times p$, where $r = p$ denotes the codimension of the code. Hence a parity-check matrix is $n \times r$ array of the form

$$H = [H_0 | H_1 | \dots | H_{n_0-1}],$$

where $n = n_0 p$ and $r = p$.

Quasi-cyclic constructions for MDPC-codes are convenient, since they show a very good performance on embedded systems. There are some improvements on the bit-flipping algorithm using quasi-cyclic MDPC-codes in an McEliece system.

Another more general construction is the random model presented by Tillich (see [37] Subsection 2.2). Using Proposition 5.3.1, he showed that for this construction the maximum column intersection is of order $\mathcal{O}\left(\frac{\log(n)}{\log \log(n)}\right)$, which is only small for small n .

Our goal is to give a construction which minimizes the maximum column intersection of a parity-check matrix as much as possible. Then the error-correction capability increases. Indeed we will show that this number is exactly 2.

⁴An $[n, k]$ -linear code C is called *quasi-cyclic*, if there is an integer n_0 such that every codeword in C shifted by n_0 places is again a codeword in C .

6.1 Parity-Check Matrix

Let us start by constructing a parity-check matrix, since we have seen that a linear code is fully characterized by those ones (see Section 2.3). The goal is to construct an MDPC-code of a certain length n and some fixed order (v, w) such that the row-weight w is roughly \sqrt{n} . We will step by step determine the parameters of the code. In the course of this section let q always be an odd prime power and consider the projective plane $PG(2, q)$ of order q . The following table gives a short review on the number of points and lines in the Desarguesian plane and their relation.

Number of lines	$q^2 + q + 1$
Number of points	$q^2 + q + 1$
Number of points on a line	$q + 1$
Number of lines through a point	$q + 1$

Table 6.1: Overview of points and lines in $PG(2, q)$

Furthermore, we have seen that a projective bundle is a set of $q^2 + q + 1$ non-degenerate conics in $PG(2, q)$ and that the configuration of points and non-degenerate conics of a projective bundle in $PG(2, q)$ is isomorphic to $PG(2, q)$ of order q , where the non-degenerate conics can be interpreted as lines. Hence the above table can be modified by replacing lines by non-degenerate conics of a fixed projective bundle in $PG(2, q)$. For the relation between conics and lines, we have seen in Section 3.4 that a line meets a non-degenerate conic in at most two points.

In the construction we use the idea of an incidence structure with a corresponding incidence matrix (see Section 3.1). We will use the incidence matrix of points and lines of the projective plane $PG(2, q)$ and the incidence matrix of non-degenerate conics of a fixed projective bundle and points of $PG(2, q)$.

For a fixed odd prime power q define a parity-check matrix H of the form

$$H = [H' | H'']$$

where H' and H'' are the following two incidence matrices (see Section 3.1 for the definition).

Let P_1, \dots, P_{q^2+q+1} denote the $q^2 + q + 1$ points of $PG(2, q)$ and let l_1, \dots, l_{q^2+q+1} denote the $q^2 + q + 1$ lines of $PG(2, q)$. The matrix H' is an incidence matrix of these points and lines in $PG(2, q)$, namely

$$H' = (h'_{ij}) = \begin{cases} 0, & \text{if } P_i \text{ does not lie on } l_j \\ 1, & \text{if } P_i \text{ lies on } l_j \end{cases}.$$

Hence, in each row and each line there are $q + 1$ one-entries. By the definition of a projective plane, every two distinct lines intersect in a unique point and each two points lie on exactly one line. Hence, the maximum column intersection s'_H of H' equals 1.

For matrix H'' we consider a fixed projective bundle of $PG(2, q)$. Let us denote the

$q^2 + q + 1$ non-degenerate conics of this bundle by c_1, \dots, c_{q^2+q+1} . Since there we view the non-degenerate conics of a projective bundle as lines of $PG(2, q)$, we define similarly matrix H'' to be the incidence matrix of the non-degenerate conics and points in $PG(2, q)$.

$$H'' = (h''_{ij}) = \begin{cases} 0, & \text{if } P_i \text{ is not contained in conic } c_j \\ 1, & \text{if } P_i \text{ is contained in conic } c_j \end{cases}.$$

Again there are $q + 1$ one-entries in each row and each column and the maximum column intersection of H'' is $s_{H''} = 1$.

To summarize, we get a binary matrix H of size $(q^2 + q + 1) \times 2(q^2 + q + 1)$ defined by the points, lines and non-degenerate conics of $PG(2, q)$ in the following way

$$H = \begin{array}{c} p_1 \\ p_2 \\ \vdots \\ p_{q^2+q+1} \end{array} \left(\begin{array}{cccc|cccc} l_1 & l_2 & \dots & l_{q^2+q+1} & c_1 & c_2 & \dots & c_{q^2+q+1} \\ \hline & & & & & & & \end{array} \right)$$

Let us denote the Desarguesian plane used for matrix H' by Π and the one used for H'' by Γ . Similar to the definition used in Section 3.5 we let $C_2(\Pi \sqcup \Gamma)$ denote the row-space of H . Then the code whose parity-check matrix is H , is given by

$$C_2(\Pi \sqcup \Gamma)^\perp = \ker(H).$$

Remark 6.1.1. $C_2(\Pi \sqcup \Gamma)^\perp$ is an MDPC-code of length $n = 2(q^2 + q + 1)$ and of type $(v, w) = (q + 1, 2(q + 1))$.

Example 6.1.2. Consider the Desarguesian plane $PG(2, q)$ with order $q = 3$. We will use the representation of points, lines and non-degenerate conics of a projective bundle in $PG(2, q)$ presented in Section 4.5, i.e. we identify the set of points with the integers modulo $N = q^2 + q + 1 = 13$ and the set of lines with the images of the perfect difference set $D = \{0, 1, 3, 9\}$ under repeated applications of the Singer cycle $S(i) = i + 1$. Thus we have

$$\begin{aligned} \mathcal{P} &= \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}, \\ \mathcal{L} &= \{\{0 + i, 1 + i, 3 + i, 9 + i\} \mid i \in \mathbb{Z}/\langle 13 \rangle\} \\ &= \{\{0, 1, 3, 9\}, \{1, 2, 4, 10\}, \{2, 3, 5, 11\}, \{3, 4, 6, 12\}, \{4, 5, 7, 0\}, \{5, 6, 8, 1\}, \{6, 7, 9, 2\}, \\ &\quad \{7, 8, 10, 3\}, \{8, 9, 11, 4\}, \{9, 10, 12, 5\}, \{10, 11, 0, 6\}, \{11, 12, 1, 7\}, \{12, 0, 2, 8\}\}. \end{aligned}$$

An incidence matrix of the points and lines in $PG(2, q)$ is then given by

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

For the projective bundle we are free to choose which of the three types we take. Let us take the inscribed bundle which is the images of $2D$ under repeated applications of the Singer cycle, by Section 4.5. Hence, the projective bundle is the following set of non-degenerate conics

$$\begin{aligned} \mathcal{B}_I &= \{\{0 + i, 2 + i, 5 + i, 6 + i\} \mid i \in \mathbb{Z}/\langle 13 \rangle\} \\ &= \{\{0, 2, 5, 6\}, \{1, 3, 6, 7\}, \{2, 4, 7, 8\}, \{3, 5, 8, 9\}, \{4, 6, 9, 10\}, \{5, 7, 10, 11\}, \{6, 8, 11, 12\}, \\ &\quad \{7, 9, 12, 0\}, \{8, 10, 0, 1\}, \{9, 11, 1, 2\}, \{10, 12, 2, 3\}, \{11, 0, 3, 4\}, \{12, 1, 4, 5\}\}, \end{aligned}$$

which leads to the following incidence matrix of points and non-degenerate conics of an inscribed bundle in $PG(2, 3)$

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

We set then H to be the concatenation of the two incidence matrices H' and H'' .

$$H = \left(\begin{array}{cccccccccccc|cccccccc} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \end{array} \right)$$

This matrix is then the parity-check matrix of a code $C_2(\Pi \sqcup \Gamma)^\perp$, where Π and Γ are the projective planes isomorphic to $PG(2, 3)$ defined by a set of points and a set of lines and non-degenerate conics of the inscribed bundle, respectively.

In this example we have chosen one of the three possible projective bundles. It would indeed be interesting to compare the three variants in how they perform. Do they all show the same error-correction performance and are they equally useful in different situations? These could be interesting questions for further studies.

We have implemented the three different types of parity-check matrices in Python using SageMath (see Listing 2 in Appendix A), in order to compute also parity-check matrices with an arbitrary parameter q .

6.2 Dimension

Earlier, in Section 3.5, we have discussed that over a prime field $GF(p)$, where p is a prime, it is rather difficult to determine the dimension of a code generated from projective planes if p divides the order q of the projective plane. This is because the structure of the projective plane is highly influencing the value of the dimension.

In our case, since we know exactly how the parity-check matrix H' of $C_2(\Pi)$ looks like and since 2 does not divide the odd order q of the projective plane, we are able to compute its dimension using the eigenvalues of $H'H'^\top$ over $GF(2)$. The following result states the exact rank of the matrix H' . Note that for H'' the same result is true, since both matrices are incidence matrices of $PG(2, q)$ of odd order q .

Lemma 6.2.1. *The eigenvalues of $H'H'^\top$ are $\lambda_1 = 0$ and $\lambda_2 = 1$, where λ_1 has multiplicity 1 and λ_2 has multiplicity $q^2 + q$. Moreover, the rank of H' is $q^2 + q$.*

Proof. First compute $H'H'^\top$. For this we note that

$$(H'H'^\top)_{ij} = \begin{cases} q + 1 & \text{if } i = j, \\ 1 & \text{else} \end{cases}.$$

In order to compute the eigenvalues we recall that for any eigenvalue λ and any non-zero vector $x \in GF(2)^{q^2+q+1}$ it holds

$$(H'H'^\top) \cdot x = \lambda \cdot x.$$

By computing $(H'H'^\top) \cdot x$ we obtain that

$$\begin{pmatrix} \vdots \\ \sum_{i=1}^{q^2+q+1} x_i \\ \vdots \end{pmatrix} = (\lambda - q) \cdot x,$$

which is equivalent to

$$\sum_{i=1}^{q^2+q+1} x_i \cdot \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} = (\lambda - q) \cdot x.$$

If $\sum_{i=1}^{q^2+q+1} x_i \neq 0$ then $x = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ is an eigenvector with eigenvalue $\lambda_1 = q^2 + 2q + 1$,

which, when reduced over $GF(2)$, is $\lambda_1 = 0$. We in addition see that this eigenvalue has multiplicity 1.

If $\sum_{i=1}^{q^2+q+1} x_i = 0$, then at least two entries of x must be reciprocal to each other. Therefore, we must have that

$$\lambda - q = 0.$$

This gives the second eigenvalue $\lambda_2 = q$, which reduced over $GF(2)$ is $\lambda_2 = 1$. Indeed λ_2 has $q^2 + q$ pairwise distinct eigenvectors. Therefore, $\lambda_2 = 1$ is of multiplicity $q^2 + q$.

Hence, we can compute the rank of $H'H'^\top$ by

$$\begin{aligned} \text{rk } H'H'^\top &= q^2 + q + 1 - \dim(\ker H'H'^\top) \\ &= q^2 + q + 1 - 1 \\ &= q^2 + q. \end{aligned}$$

Therefore H' must have rank $q^2 + q$ as well, otherwise if it has full rank, H'^\top has full rank too and then $H'H'^\top$ has full rank which, as shown above, is not the case. \square

With the help of the above Lemma 6.2.1 we are able to explicitly determine the dimension of the code we have constructed.

Proposition 6.2.2. *An MDPC-code $C_2(\Pi \sqcup \Gamma)^\perp$ has dimension $q^2 + q + 2$.*

Proof. In order to determine the dimension of the code, we need to compute the rank of a parity-check matrix $H = (H'|H'')$. Since H is of size $(q^2 + q + 1) \times 2(q^2 + q + 1)$, we can already say that the rank of H is at most $q^2 + q + 1$. From the above lemma we have that both H' and H'' have rank $q^2 + q$, which gives us a lower bound on the rank of H .

H has full rank $q^2 + q + 1$ if and only if there exists no element in the left-kernel, i.e. if there is no non-zero vector $x \in GF(2)^{q^2+q+1}$ such that

$$x^\top \cdot H = 0^\top \text{ over } GF(2).$$

Indeed, if x is the all-one vector then this equation is satisfied. Hence, there is an element in the cokernel which implies that H cannot have full rank and we conclude that $\dim C_2(\Pi \sqcup \Gamma)^\perp = q^2 + q + 2$. \square

As a consequence we have parametrized our MDPC-Code $C_2(\Pi \sqcup \Gamma)^\perp$ which is a $[2(q^2 + q + 1), q^2 + q + 2]_2$ -linear code.

6.3 Minimum Distance

Earlier we have seen that the minimum distance of a code is relevant to determine how many errors can be corrected and detected (see Proposition 2.2.3). In favour to

be able to correct and detect errors, we want to find an estimate on the minimum distance.

To give more detailed information on the error-correction and -detection capability we want to find lower and upper bounds on the minimum distance.

Let us first prove the following lemma which we will then use to find bounds on the minimum distance.

Lemma 6.3.1. *Let $C_2(\Pi \sqcup \Gamma)^\perp$ be an MDPC-code with parity-check matrix H as constructed. Consider an arbitrary but minimal set $S = \{l_1, \dots, l_r, c_1, \dots, c_s\}$ of linearly dependent columns of H , where l_i are some columns corresponding to lines and c_i are some columns corresponding to non-degenerate conics of a projective bundle of $PG(2, q)$. Then the following equality holds*

$$\left(\bigcup_{i=1}^r l_i \right) \cup \left(\bigcup_{i=1}^s c_i \right) = \left(\bigcup_{i < j} l_i \cap l_j \right) \cup \left(\bigcup_{i < j} c_i \cap c_j \right) \cup \left(\bigcup_{i,j} l_i \cap c_j \right).$$

Proof. We will show that the two sets are each contained in the other.

First observe, that the inclusion from right to left is clearly given for all of the columns of H . So it remains to show the inclusion from left to right.

Let $S = \{l_1, \dots, l_r, c_1, \dots, c_s\}$ be a minimal set of linearly dependent columns of H . Since we work over $GF(2)$, we get

$$l_1 + \dots + l_r + c_1 + \dots + c_s = 0, \quad (6.1)$$

where 0 here denotes the zero-vector.

Let p_j be the j^{th} point of $PG(2, q)$ and assume that p_j lies on one of these lines or conics, i.e.

$$p_j \in \left(\bigcup_{i=1}^r l_i \right) \cup \left(\bigcup_{i=1}^s c_i \right).$$

Without loss of generality, we can say that p_j lies on the line $l_i \in S$. Then the j^{th} entry of l_i is a 1. By linear dependence and (6.1) there must be at least one other column having a one-entry at position j . Hence for some $l_k, c_{k'} \in S \setminus \{l_i\}$, we have that

$$p_j \in l_k \quad \text{or} \quad p_j \in c_{k'}.$$

Hence we conclude

$$p_j \in \left(\bigcup_{i < k} l_i \cap l_k \right) \quad \text{or} \quad p_j \in \left(\bigcup_{i, k'} l_i \cap c_{k'} \right).$$

Analogously if $p_j \in c_i$ we get that

$$p_j \in \left(\bigcup_{i < k'} c_i \cap c_{k'} \right) \quad \text{or} \quad p_j \in \left(\bigcup_{i, k'} l_i \cap c_{k'} \right).$$

Therefore, we can deduce the inclusion from left to right. \square

With the help of this Lemma 6.3.1 we are able to find a lower bound on the minimum distance of $C_2(\Pi \sqcup \Gamma)^\perp$.

Proposition 6.3.2. *Let d denote the minimum distance of the code $C_2(\Pi \sqcup \Gamma)^\perp$. Then the following estimate holds*

$$\left\lfloor \frac{2q+4}{3} \right\rfloor + 1 \leq d.$$

Proof. To prove the lower bound for the minimum distance d we will use Lemma 6.3.1 and Theorem 2.3.11.

Let $S = \{l_1, \dots, l_r, c_1, \dots, c_s\}$ be a set of $r + s$ columns of the parity-check matrix H , where l_1, \dots, l_r are columns corresponding to lines and c_1, \dots, c_s are columns corresponding to conics. From the above lemma we get that if S is a linearly dependent set, then

$$\left| \underbrace{\left(\bigcup_{i=1}^r l_i \right) \cup \left(\bigcup_{i=1}^s c_i \right)}_{=:A} \right| = \left| \underbrace{\left(\bigcup_{i<j} l_i \cap l_j \right) \cup \left(\bigcup_{i<j} c_i \cap c_j \right) \cup \left(\bigcup_{i,j} l_i \cap c_j \right)}_{=:B} \right|.$$

The idea is to find two bounds g and h depending on r and s such that

$$g(r, s) \leq |A| = |B| \leq h(r, s).$$

Then, by the above lemma, if

$$h(r, s) - g(r, s) < 0,$$

certainly any set of columns obtained selecting r columns from the lines and s columns from the non-degenerate conics can not be linearly dependent.

So let us find a lower bound on $|A|$ first.

We can rewrite A as the union of two sets $A_1 = (\bigcup_{i=1}^r l_i)$ and $A_2 = (\bigcup_{i=1}^s c_i)$. Then we have that

$$|A| = |A_1| + |A_2| - |A_1 \cap A_2|.$$

We get the following estimates for the three parts.

$$\begin{aligned} |A_1| &\geq \sum_{i=1}^r |l_i| - \sum_{i<j} |l_i \cap l_j| = r(q+1) - \frac{r(r-1)}{2} \\ |A_2| &\geq \sum_{i=1}^s |c_i| - \sum_{i<j} |c_i \cap c_j| = s(q+1) - \frac{s(s-1)}{2} \\ |A_1 \cap A_2| &= \left| \bigcup_{i,j} l_i \cap c_j \right| \leq \sum_{i,j} |l_i \cap c_j| \leq \sum_{i,j} 2 = 2rs. \end{aligned}$$

Hence we obtain the following lower bound for the cardinality of the set A .

$$|A| \geq \frac{(r+s)(2q+3) - (r+s)^2 - 2rs}{2} =: g(r, s).$$

For the upper bound on $|B|$ we first write B as the union $B = B_1 \cup B_2 \cup B_3$, where $B_1 = (\bigcup_{i < j} l_i \cap l_j)$, $B_2 = (\bigcup_{i < j} c_i \cap c_j)$ and $B_3 = (\bigcup_{i, j} l_i \cap c_j)$. To find an upper bound, we can use, that

$$|B| \leq |B_1| + |B_2| + |B_3|$$

and find upper bounds for each of the three cardinalities. Indeed, by the same computations as above, we obtain

$$\begin{aligned} |B_1| &\leq \frac{r(r-1)}{2}, \\ |B_2| &\leq \frac{s(s-1)}{2}, \\ |B_3| &\leq 2rs. \end{aligned}$$

Thus we get

$$|B| \leq \frac{(r+s)^2 - (r+s) + 2rs}{2} =: h(r, s).$$

Now if S is a set of linearly dependent columns, then

$$g(r, s) \leq h(r, s),$$

which is equivalent to saying that

$$f(r, s) := h(r, s) - g(r, s) = (r+s)^2 - (r+s)(q+2) + 2rs \geq 0.$$

Therefore if $f(r, s) < 0$, any set of columns obtained selecting r columns from the lines and s columns from the non-degenerate conics can not be linearly dependent.

We now want to find the maximal possible value of $(r+s)$ such that $f(r, s) < 0$ is satisfied for every r and s with fixed sum $r+s$.

If $r+s = T$ is a fixed value then $2rs$ is maximal when $r = s$. This means that

$$f(r, s) \leq (s+s)^2 - (s+s)(q+2) + 2s^2,$$

and hence we want to find the maximal value of s such that

$$(s+s)^2 - (s+s)(q+2) + 2s^2 < 0,$$

Indeed, this is the case when

$$2s(3s - (q+2)) < 0.$$

This is satisfied when

$$\begin{aligned} \text{either } s < 0 \text{ and } s > \frac{q+2}{3}, \\ \text{or } s > 0 \text{ and } s < \frac{q+2}{3}. \end{aligned}$$

Since $s > 0$ is given, only the latter case is possible. Hence, $s < \frac{q+2}{3}$ and therefore the maximal value T for $r + s$ satisfying $f(r, s) < 0$ is

$$T < \left\lfloor \frac{2q+4}{3} \right\rfloor.$$

Therefore any set of linearly dependent columns of H has cardinality at most $\lfloor \frac{2q+4}{3} \rfloor$. Hence by Theorem 2.3.11 we conclude that the minimum distance d is at least $\lfloor \frac{2q+4}{3} \rfloor + 1$. \square

In Chapter 3 we have seen that there are two other possibilities to examine a lower bound on the minimum distance of the code. Depending on the situation, one or the other bound is more useful and accurate. We would like to apply the two Theorems 3.5.2 and 3.5.3 to our code $C_2(\Pi \sqcup \Gamma)^\perp$. In order to do so we need to compute the eigenvalues of HH^\top over the reals, where H is the parity-check matrix of $C_2(\Pi \sqcup \Gamma)^\perp$.

In the computation of the dimension of $C_2(\Pi \sqcup \Gamma)^\perp$ we saw that the matrix $H'H'^\top$ has the eigenvalues $\lambda_1 = (q+1)^2$ and $\lambda_2 = q$ over the real numbers. Indeed, it holds that

$$HH^\top = (H'|H'') \cdot (H'|H'')^\top = \begin{pmatrix} 2(q+1) & 2 & \dots & 2 \\ 2 & 2(q+1) & 2 & \dots \\ \vdots & \ddots & \ddots & \vdots \\ 2 & \dots & 2 & 2(q+1) \end{pmatrix} = 2 \cdot (H'H'^\top). \quad (6.2)$$

This implies then that for any $x \in \mathbb{R}^{q^2+q+1}$

$$(HH^\top) \cdot x = 2(H'H'^\top) \cdot x = 2 \cdot \lambda_1 x = (2\lambda_1) \cdot x,$$

and analogously for λ_2 , since these are the eigenvalues of $H'H'^\top$. Therefore the two eigenvalues of HH^\top over the real numbers are

$$\mu_1 = 2(q+1)^2 \quad \text{and} \quad \mu_2 = 2q.$$

Remark 6.3.3. Since q is an odd prime power and every two points are lying on exactly one common line, any pair of two distinct columns of a parity-check matrix H are linearly independent. Hence, by Theorem 2.3.11, we can guarantee that the minimum distance of such an MDPC-code $C_2(\Pi \sqcup \Gamma)^\perp$ is at least 3.

Having the two eigenvalues of HH^\top we are able to use Theorem 3.5.2 and Theorem 3.5.3, respectively. Both theorems lead to exactly the same result for the minimum distance d of $C_2(\Pi \sqcup \Gamma)^\perp$, i.e.

$$d \geq 2.$$

Hence, Tanner's results are not really useful for a code $C_2(\Pi \sqcup \Gamma)^\perp$ that we have constructed. Indeed, we have discussed in Chapter 3, that if $\mu_2 \geq 2m$ the bound provided by Theorem 3.5.2 is vacuous. In our case we indeed have $\mu_2 \leq 2m$, but very close to $2m$ which possibly could be a reason for the loss of accuracy. Therefore, the combinatorial approach presented in Proposition 6.3.2 is definitely the better choice.

In order to find an upper bound for the minimum distance, one could use the minimum distance of a code $C_2(\Pi)^\perp$, which is not very tight. Instead, for binary linear codes we have seen that the Griesmer bound, Theorem 2.3.13, provides an appropriate upper bound.

6.4 Error-Correction Capability

We have constructed $[2(q^2 + q + 1), q^2 + q + 2]_2$ -linear MDPC-codes $C_2(\Pi \sqcup \Gamma)^\perp$ of column weight $v = (q + 1)$ for some fixed odd prime power q . We are now interested in the error-correction capability of these codes.

Using Proposition 2.2.3 and the lower bound on the minimum distance of an MDPC-code $C_2(\Pi \sqcup \Gamma)^\perp$ presented in Proposition 6.3.2, we can deduce that an MDPC-code $C_2(\Pi \sqcup \Gamma)^\perp$ has the following error-correction and error-detection properties:

1. If $\left\lceil \frac{2q+4}{3} \right\rceil$ is odd, then $C_2(\Pi \sqcup \Gamma)^\perp$ can correct up to $\frac{\left\lceil \frac{2q+4}{3} \right\rceil - 1}{2}$ errors.
2. $C_2(\Pi \sqcup \Gamma)^\perp$ can detect up to $\left\lceil \frac{2q+4}{3} \right\rceil - 1$ errors.
3. $C_2(\Pi \sqcup \Gamma)^\perp$ can correct up to $\left\lceil \frac{2q+4}{3} \right\rceil - 1$ erasures.

Recall that using an $[n, k]$ -linear MDPC-code of column weight v and parity-check matrix H and maximum column intersection s we can correct errors of weight at most $\lfloor \frac{v}{2s} \rfloor$ by performing only one round of the bit-flipping algorithm (see Proposition 5.3.1).

Lemma 6.4.1. *Let $C_2(\Pi \sqcup \Gamma)^\perp$ be an MDPC-code with parity-check matrix $H \in GF(2)^{(q^2+q+1) \times 2(q^2+q+1)}$ constructed as above. Then the maximum column intersection s_H of H is at most 2.*

Proof. From the construction of H we have that H consists of two matrices H' and H'' which are the incidence matrices of points and lines and points and non-degenerate conics of a projective bundle in $PG(2, q)$, respectively, since any two points are incident to exactly one common line and one common non-degenerate conic. Hence both matrices H' and H'' have a maximum column intersection equal to 1. Since every line is intersecting a conic in at most 2 points, the maximum column intersection of the matrix H is $s_H \leq 2$. \square

Using Lemma 6.4.1 and Proposition 5.3.1, we can directly derive the maximal number of errors that can be corrected within one round of the bit-flipping decoding algorithm.

Corollary 6.4.2. *Consider the MDPC-code $C_2(\Pi \sqcup \Gamma)^\perp$, with parity-check matrix as constructed before. After performing one round of bit-flipping on H we can correct errors of weight up to $\lfloor \frac{q+1}{4} \rfloor$, which is roughly $\sqrt{\frac{n}{32}}$.*

When compared to Tillich's random construction of MDPC-codes, presented in [24], we can guarantee an improvement on the number of errors that can be corrected

within one round of the bit-flipping decoding algorithm.

In addition to this result, we have implemented the bit-flipping algorithm in Python using SageMath (see Listing 3 in Appendix A). Our goal was not only to verify the bound found in Corollary 6.4.2 for the family of MDPC-codes constructed in Section 6.1, but also to test if we could possibly correct more than this amount of errors within one round of the bit-flipping algorithm. In order to do so, we use the implementation of the three types of parity-check matrices which can be found in Listing 2. From Display (6.2), a parity-check matrix of the family of MDPC-codes $C_2(\Pi \sqcup \Gamma)^\perp$ forms a generator matrix of a subcode of $C_2(\Pi \sqcup \Gamma)^\perp$. Hence, the codewords of this subcode are also codewords of $C_2(\Pi \sqcup \Gamma)^\perp$.

Since we want to check how many errors we can correct within one round of the bit-flipping decoding algorithm, we need not only a parity-check matrix, but also a received word, which is technically a codeword of $C_2(\Pi \sqcup \Gamma)^\perp$ plus some error-vector of a given weight. Since the algorithm does not depend on the codeword, we have chosen an arbitrary row of H which represents a codeword of $C_2(\Pi \sqcup \Gamma)^\perp$. To this randomly chosen codeword of $C_2(\Pi \sqcup \Gamma)^\perp$ we have added an error-vector of different weights presented below. The error-vector is constructed in the following way.

Initially, we have generated a zero-vector of length $q^2 + q + 1$. Depending on the weight $wt(e)$, that the error-vector e should have, we have chosen $wt(e)$ random positions which will be turned into a one-entry. The construction of the error-vector is therefore pseudo-random and hence useful for an experiment. The SageMath code can be found in Listing 4.

For the test, we have chosen q to be every odd prime power between 5 and 25, i.e.

$$q \in \{5, 7, 9, 11, 13, 17, 19, 23, 25\}.$$

For each value of q and each error-weight, we ran the algorithm 1000 times in order to provide an accurate result. Additionally, we have examined in all these test all the three types of parity-check matrices according to the three types of projective bundles found in Section 4.5. The SageMath code for the test loops can be found in Appendix A, Listing 4.

Firstly, in order to approve the bound found in Corollary 6.4.2, we have added to some arbitrarily chosen codeword a random error-vector whose weight is exactly this bound. In fact, it turned out that in each of the 1000 tests all the received words containing an error of weight $\lfloor \frac{q+1}{4} \rfloor$ have been decoded completely into a codeword of $C_2(\Pi \sqcup \Gamma)^\perp$.

Secondly, we have increased the weight of the error by 1, i.e. the error-vector has now a weight of $\lfloor \frac{q+1}{4} \rfloor + 1$. After running the algorithm 1000 times for each of the three types of parity-check matrices and for each value of q , we have computed the percentage number of successfully decoded words which can be found in the following table.

q	inscribed bundle	circumscribed bundle	self-polar bundle
5	53.5%	53.5%	53.5%
7	4.2 %	3.9%	3.9%
9	75.9%	75.4%	76.0%
11	43.8%	42.8%	42.1%
13	91.9%	91.3%	90.5%
17	96.0%	96.6%	96.0%
19	91.5%	91.6%	91.3%
23	97.4%	98.0%	97.8%
25	98.9%	98.9%	100%

Table 6.2: Probability to decode a received word of $\lfloor \frac{q+1}{4} \rfloor + 1$ errors correctly after one round of the bit-flipping decoding algorithm.

From the table we realize that for the prime $q = 7$ the probability that the decoding fails is very high. This might be due to the fact that we exceed the error-correcting radius. Recall from the error-correction properties presented at the beginning of this section, for $q = 7$ we can correct up to 2 errors. Since $\lfloor \frac{q+1}{4} \rfloor + 1$ for $q = 7$ is already equal to 3, we exceed the number of errors that can be corrected. Furthermore, when computing $\frac{q+1}{4}$ for $q = 7$, it is already an integer. When compared to $q = 5$, for which $\frac{q+1}{4}$ is non-integer and will be rounded down, we observe a decrease in the performance. The fact, that for $q = 5$ the fraction is non-integer might also have an influence on the error-correction performance. The same can actually be observed for $q = 11$ or $q = 19$, whose performance slightly suffers when compared to $q = 9$ or $q = 17$ respectively.

Since especially the higher primes and prime powers showed a high rate of success, we have decided to increase the error-weight again by one and to see how the performances change. Since $q = 7$ already showed a bad performance, we have excluded this case in the next few tests. Running again 1000 tests for an error-weight of $\lfloor \frac{q+1}{4} \rfloor + 2$ we obtain the following results on the percentage success of each type.

q	inscribed bundle	circumscribed bundle	self-polar bundle
5	2.9%	2.9%	2.9%
9	6.1%	5.0%	5.9%
11	4.3%	4.8%	4.8%
13	16.9%	17.5%	17.0%
17	59.4%	58.6%	57.7%
19	45.1%	45.6%	46.6%
23	78.0%	80.1%	77.7%
25	95.8%	95.0%	94.5%

Table 6.3: Probability to decode a received word of $\lfloor \frac{q+1}{4} \rfloor + 2$ errors correctly after one round of the bit-flipping decoding algorithm.

From the above table, we can clearly see that for $q = 5, 9, 11, 13$ the probability of decoding a received word containing $\lfloor \frac{q+1}{4} \rfloor + 2$ errors after one round of the bit-flipping decoding algorithm is very low. This is due to the fact, that for these values of q the weight of the error is higher than the theoretical number of errors that can be corrected, i.e. since

$$\left\lfloor \frac{q+1}{4} \right\rfloor + 2 \geq \frac{\left\lceil \frac{2q+4}{3} \right\rceil - 1}{2}.$$

Still the performance for $q = 19$ is slightly smaller than the one for $q = 17$. Furthermore, for $q = 25$ the error-correction rate is still very high.

Lastly, we have decided to increase again the error-weight by one to see how the performances change. Similarly to the case before, we will only focus on the cases where $q = 17$ or higher. Again after 1000 test loops for each parameter we observe the following results.

q	inscribed bundle	circumscribed bundle	self-polar bundle
17	7.9%	6.4%	7.2%
19	10.4%	9.8%	11.3%
23	31.1%	30.4%	31.7%
25	75.8%	74.0%	71.1%

Table 6.4: Probability to decode a received word of $\lfloor \frac{q+1}{4} \rfloor + 3$ errors correctly after one round of the bit-flipping decoding algorithm.

Interestingly, in this situation the performance for $q = 19$ is now better than the one for $q = 17$, but this again might be due to the fact, that the general amount of errors which can be corrected is exceeded when choosing $q = 17$. Also the performance for

$q = 23$ has significantly suffered when compared to the performance in the previous setting. Only for $q = 25$ the rate of success is still high.

Since we were curious on finding the limit of error-correction capacities for the presented values for q , we ran the algorithm another 1000 times only for the value $q = 25$ with an error-weight of $\lfloor \frac{q+1}{4} \rfloor + 4$. It turned out that the rate of success has decreased rapidly, which can be seen in the last table below.

q	inscribed bundle	circumscribed bundle	self-polar bundle
25	35.2%	32.6%	36.9%

Table 6.5: Probability to decode a received word of $\lfloor \frac{q+1}{4} \rfloor + 4$ errors correctly after one round of the bit-flipping decoding algorithm.

Generally, we can conclude from the results above that all the three types of MPDC-codes $C_2(\Pi \sqcup \Gamma)^\perp$ or rather the three types of parity-check matrices according to the three different types of projective bundles in $PG(2, q)$ show more or less the same performance. There is no significant difference between the single types. Also from these experiments we observe that for bigger odd prime powers q with a high probability we are able to correct even more than $\lfloor \frac{q+1}{4} \rfloor$ errors within one round of the bit-flipping decoding algorithm.

Chapter 7

Conclusion

To conclude this thesis, we will summarize the main results here and discuss some interesting topics for further studies.

In this Master thesis we present a new construction for MDPC-codes which have a small maximum column intersection and hence a good error-correction performance after one round of the bit-flipping decoding algorithm.

After introducing the general framework of coding theory and finite geometry, we focus on a concrete object in finite geometry: the projective bundles in the Desarguean plane $PG(2, q)$. These projective bundles play a central role in the construction. We see that there exist in total three distinct types of projective bundles in $PG(2, q)$, where two of them only exist when q is an odd prime power. We study the algebraic structure of the bundles, where we made use of perfect difference sets.

With the help of this knowledge, we are able to construct a parity-check matrix, which is the incidence matrix of points and lines or points and non-degenerate conics of a projective bundle, respectively. We use the fact, that a line in $PG(2, q)$ intersects a non-degenerate conic in at most two points, to make sure that the parity-check matrix presented has a small maximum column intersection.

We improve the error-correction performance within one round of the bit-flipping decoding algorithm when compared to the random construction presented by Tillich in [37]. In addition, the number of errors that can be corrected is deterministic and not probabilistic. Furthermore, we analyse the performance more detailed by checking whether we are able to correct even more errors. This is in fact possible for larger odd prime powers q .

Concerning the bit-flipping decoding algorithm for MDPC-codes, future studies could address following questions: Where does the decoding algorithm fail? Are there any issues when using a parity-check matrix constructed from finite geometry? Since for LDPC-codes the decoding algorithm faces some problems with four-cycles it could possibly happen with MDPC-codes too. Since for MDPC-codes the main analysis on decoding algorithms are based on the bit-flipping decoding algorithm and since the bit-flipping decoding algorithm only works for a binary alphabet, one could emphasize more on decoding algorithms, such as believe propagation.

Even though in this thesis we do not focus on the cryptographic point of view, another topic of great interest would be, to implement this construction for instance in a McEliece cryptosystem variant for MDPC-codes. After that, one could analyse the security level of such a cryptosystem.

Appendix A

Sage Functions

We will provide here the Sage functions used in Section 6.4.

```
from math import factorial
from sage.all import *

def perfect_difference_set(q):
    # The function computes a reduced perfect difference set of q+1
    ↪ integers.
    # :param q: odd prime power
    # :return: list representing the perfect difference set

    base_field = GF(q, 'a')
    extension = base_field.extension(3)
    gen = extension.gen()

    powers = [0, 1]
    for power in range(2, q ** 2 + q + 1):
        for k_1 in base_field:
            if k_1 != 0:
                for k_2 in base_field:
                    if k_2 != 0:
                        if 1 + k_1 * gen + k_2 * (gen ** power) == 0:
                            powers.append(power)

    differences = []
    for i in range(0, len(powers)):
        for j in range(i + 1, len(powers)):
            if ((powers[i] - powers[j]) and (powers[j] - powers[i])) not in
                ↪ differences:
                differences.append(abs(powers[i] - powers[j]))
            else:
                print(powers), print("\nis no perfect difference set")

    if len(differences) == q * (q + 1) / 2:
        print(powers), print("is a perfect difference set\n")

    return powers
```

Listing 1: Implementation of perfect difference sets of $q + 1$ elements.

```

from sage.all import *

def incidence_inscribed(q, diff_set):
    # Computes the incidence matrix of points of PG(2,q) and lines / conics
    ↪ of an inscribed proj. bundle
    # :param q: odd prime power
    # :param diff_set: list, representing a perfect difference set.
    # :return: Matrix of size  $(q^2+q+1) \times (2(q^2+q+1))$ 

    n = q ** 2 + q + 1
    lines = [0] * n
    for i in range(0, n):
        if i in diff_set:
            lines[i] = 1
    in_set = []
    in_bundle = [0] * n
    for i in range(0, q + 1):
        in_set.append(2 * diff_set[i] % n)
    for i in range(0, n):
        if i in in_set:
            in_bundle[i] = 1
    H = block_matrix([[matrix.circulant(lines).transpose(),
    ↪ matrix.circulant(in_bundle).transpose()]])
    return H

def incidence_circumscribed(q, diff_set):
    # Computes the incidence matrix of points of PG(2,q) and lines / conics
    ↪ of a circumscribed proj. bundle
    # :param q: odd prime power
    # :param diff_set: list, representing a perfect difference set.
    # :return: Matrix of size  $(q^2+q+1) \times (2(q^2+q+1))$ 

    n = q ** 2 + q + 1
    lines = [0] * n
    for i in range(0, n):
        if i in diff_set:
            lines[i] = 1
    circ_set = []
    circ_bundle = [0] * n
    for i in range(0, q + 1):
        circ_set.append((-1) * diff_set[i] % n)
    for i in range(0, n):
        if i in circ_set:
            circ_bundle[i] = 1
    H = block_matrix([[matrix.circulant(lines).transpose(),
    ↪ matrix.circulant(circ_bundle).transpose()]])

    return H

def incidence_selfpolar(q, diff_set):

```

```

# Computes the incidence matrix of points of PG(2,q) and lines / conics
↪ of an self-polar proj. bundle
# :param q: odd prime power
# :param diff_set: list, representing a perfect difference set.
# :return: Matrix of size (q^2+q+1)x(2(q^2+q+1))

n = q ** 2 + q + 1
lines = [0] * n
for i in range(0, n):
    if i in diff_set:
        lines[i] = 1
self_set = []
self_bundle = [0] * n
inv_two = inverse_mod(2, n)
for i in range(0, q + 1):
    self_set.append(inv_two * diff_set[i] % n)
for i in range(0, n):
    if i in self_set:
        self_bundle[i] = 1
H = block_matrix([[matrix.circulant(lines).transpose(),
↪ matrix.circulant(self_bundle).transpose()]])

return H

```

Listing 2: Computation of the three types of parity-check matrices for the MDPC-codes constructed in Section 6.1.

```

import numpy as np

from sage.all import *

def bit_flipping(y, H):
    # This function defines one round of the bit-flipping decoding
    ↪ algorithm for MDPC-codes.
    # :param y: vector of length 2(q^2+q+1) -> received word containing
    ↪ some errors
    # :param H: matrix of size (q^2+q+1)x(2(q^2+q+1)) -> parity-check
    ↪ matrix
    # :return: "decoded" word after one round of the bit-flipping algorithm

    r = H.nrows()
    n = len(H[1])

    # n_j[j] : number of non-zero entries in a column j
    n_j = []

    # u_j[j] : number of unsatisfied check equations in column j
    u_j = []

    summands = []

```

```

for i in range(0, r):
    summand_i = 0
    for k in range(0, n):
        summand_i = summand_i + H[i, k] * y[k]
    summand_i = summand_i % 2
    summands.append(summand_i)

for j in range(0, n):
    counter_n = 0
    for i in range(0, r):
        if H[i, j] == 1:
            counter_n += 1
    n_j.append(counter_n) # number of non-zero entries in a column j.

for j in range(0, n):
    counter_u = 0
    for i in range(0, r):
        if (H[i, j] == 1) and (summands[i] == 1):
            counter_u += 1
    u_j.append(counter_u) # number of unsatisfied p.c. equations in a
    ↪ column j.
for j in range(0, n):
    if u_j[j] > n_j[j] / 2: # flipping condition.
        y[j] = 1 - y[j]
return y

```

Listing 3: One round of bit-flipping decoding algorithm due to Algorithm 1.

```

from sage.all import *

import random
import incidence
import bit_flipping

def test_loop(q, error_increase, diff_set, loop_amount):
    # This function tests if a word containing an error which exceeds the
    ↪ theoretical weight of  $\lfloor (q+1)/4 \rfloor$ , from
    # Tillich's Result, can still be corrected within one round of the
    ↪ bit-flipping decoding algorithm.
    # param q: odd prime power
    # param error_increase: integer, amount of error weight increase
    # param diff_set: list, representing a perfect difference set.
    # return: boolean; true if the decoded word is a codeword and false if
    ↪ it is not.

    # The three different types of incidence matrices
    H_in = incidence.incidence_inscribed(q, diff_set)
    H_circ = incidence.incidence_circumscribed(q, diff_set)
    H_self = incidence.incidence_selfpolar(q, diff_set)

    # Creation of .csv file saving the values

```

```

filename = "q_{}_errinc_{}_test.csv".format(q, error_increase)

# Theoretical number of errors that can be corrected within one round
↪ of BF
weight_errors = floor((q + 1) / 4)

count_in = 0
count_circ = 0
count_self = 0

for amount in range(0, loop_amount):

    # generation of the errorvector
    error = vector([0] * len(H_in[1]))

    # for amount in range(0, loop_amount):
    errorpositions = []

    # Choose random positions of the error-vector to be 1.
    for x in range(0, weight_errors + error_increase):
        errorpositions.append(random.randint(0, len(H_in[1]) - 1))

    # Add 1-entries to the error vector
    for i in errorpositions:
        error[i] = 1

    # received word with some error of weight "weight_errors"
    y_in = (H_in[random.randint(0, H_in.nrows() - 1)] + error) % 2
    y_circ = (H_circ[random.randint(0, H_circ.nrows() - 1)] + error) %
    ↪ 2
    y_self = (H_self[random.randint(0, H_self.nrows() - 1)] + error) %
    ↪ 2

    # Run the bit-flipping decoding algorithm once for each of the
    ↪ received words
    y_in = bit_flipping.bit_flipping(y_in, H_in)
    y_circ = bit_flipping.bit_flipping(y_circ, H_circ)
    y_self = bit_flipping.bit_flipping(y_self, H_self)

    # Check if "decoded" word is a codeword
    syndrome_in = []
    syndrome_circ = []
    syndrome_self = []
    for i in range(0, H_in.nrows()):
        sum_row_in = 0
        sum_row_circ = 0
        sum_row_self = 0
        for j in range(0, len(H_in[1])):
            sum_row_in = (sum_row_in + H_in[i, j] * y_in[j]) % 2
            sum_row_circ = (sum_row_circ + H_circ[i, j] * y_circ[j]) %
            ↪ 2
            sum_row_self = (sum_row_self + H_self[i, j] * y_self[j]) %
            ↪ 2

```

```

syndrome_in.append(sum_row_in)
syndrome_circ.append(sum_row_circ)
syndrome_self.append(sum_row_self)

if syndrome_in == [0] * H_in.nrows():
    codeword_in = True
    count_in += 1
    # print("The decoded word y_in is indeed a codeword.")
else:
    codeword_in = False
    # print("There are still errors in y_in ! Iterate again.")

if syndrome_circ == [0] * H_circ.nrows():
    codeword_circ = True
    count_circ += 1
    # print("The decoded word y_circ is indeed a codeword.")
else:
    codeword_circ = False
    # print("There are still errors in y_circ ! Iterate again.")

if syndrome_self == [0] * H_self.nrows():
    codeword_self = True
    count_self += 1
    # print("The decoded word y_self is indeed a codeword.")
else:
    codeword_self = False
    # print("There are still errors in y_self ! Iterate again.")

with open(filename, "a+") as file:
    file.write("{} , {} , {} \n".format(codeword_in, codeword_circ,
    ↪ codeword_self))

with open(filename, "a+") as file:
    file.write("{} , {} , {} \n".format(count_in/loop_amount * 100,
    ↪ count_circ/loop_amount * 100, count_self/loop_amount * 100))

```

Listing 4: Test loop which runs one round of the bit-flipping decoding algorithm on each of the three parity-check matrices presented and an arbitrarily chosen word of given error-weight.

Bibliography

- [1] E. F. Assmus and J. D. Key. *Designs and their Codes*. Number 103. Cambridge University Press, 1994.
- [2] R. D. Baker, J. M. N. Brown, G. L. Ebert, J. C. Fisher, et al. Projective bundles. *Bulletin of the Belgian Mathematical Society-Simon Stevin*, 1(3):329–336, 1994.
- [3] M. Baldi, M. Bodrato, and F. Chiaraluce. A new analysis of the McEliece cryptosystem based on QC-LDPC codes. In *International Conference on Security and Cryptography for Networks*, pages 246–262. Springer, 2008.
- [4] S. Ball and Z. Weiner. An introduction to finite geometry. *Preprint*, 162, 2011.
- [5] T. Berner. Codierungstheorie. *University of Zurich available at https://www.math.uzh.ch/fvm/fileadmin/documents/mitschriften/codierungstheorie/M_FS08.pdf*, 2008.
- [6] D. J. Bernstein. Post-quantum cryptography. *Encyclopedia of Cryptography and Security*, pages 949–950, 2011.
- [7] R. Bruck. Circle geometry in higher dimensions. In *A survey of combinatorial theory*, pages 69–77. Elsevier, 1973.
- [8] P. J. Cameron, Q. Mary, and W. C. U. of London). *Projective and polar spaces*, 1992.
- [9] P. Dembowski. *Finite Geometries: Reprint of the 1968 edition*. Springer Science & Business Media, 2012.
- [10] R. Gallager. Low-density parity-check codes. *IRE Transactions on information theory*, 8(1):21–28, 1962.
- [11] D. G. Glynn. *Finite projective planes and related combinatorial systems*. PhD thesis, University of Adelaide Adelaide, 1978.
- [12] J. H. Griesmer. A bound for error-correcting codes. *IBM Journal of Research and Development*, 4(5):532–542, 1960.
- [13] V. Guruswami. Iterative decoding of low-density parity check codes (a survey). *arXiv preprint cs/0610022*, 2006.
- [14] V. Guruswami, A. Rudra, and M. Sudan. Essential coding theory. *Draft available at <http://www.cse.buffalo.edu/atri/courses/coding-theory/book>*, 2012.

- [15] H. Halberstam and R. Laxton. Perfect difference sets. *Glasgow Mathematical Journal*, 6(4):177–184, 1964.
- [16] M. Hall Jr et al. Cyclic projective planes. *Duke Mathematical Journal*, 14(4):1079–1090, 1947.
- [17] J. Hirschfeld. *Projective geometries over finite fields*. Oxford University Press, 1998.
- [18] A. Janoska. MDPC decoding algorithms and their impact on the McEliece cryptosystem. In *2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 1085–1089. IEEE, 2018.
- [19] G. J. Järnefelt and P. E. Kustaanheimo. An observation on finite geometries. *Den 11 te Skandinaviske Matematikerkongress*, pages 166–182, 1949.
- [20] Y. Lindell. Introduction to Coding Theory Lecture notes. *Department of Computer Science Bar-Ilan University, Israel January, 25, 2010*.
- [21] Z. Liu and D. A. Pados. LDPC codes from generalized polygons. *IEEE transactions on information theory*, 51(11):3890–3898, 2005.
- [22] G. A. Margulis. Explicit constructions of graphs without short cycles and low density codes. *Combinatorica*, 2(1):71–78, 1982.
- [23] R. J. McEliece. A public-key cryptosystem based on algebraic. *Coding Thv*, 4244:114–116, 1978.
- [24] R. Misoczki, J.-P. Tillich, N. Sendrier, and P. S. Barreto. MDPC-McEliece: New McEliece variants from moderate density parity-check codes. In *2013 IEEE international symposium on information theory*, pages 2069–2073. IEEE, 2013.
- [25] C. Monico, J. Rosenthal, and A. Shokrollahi. Using low density parity check codes in the McEliece cryptosystem. In *2000 IEEE International Symposium on Information Theory (Cat. No. 00CH37060)*, page 215. IEEE, 2000.
- [26] J. Pearl. *Reverend Bayes on inference engines: A distributed hierarchical approach*. Cognitive Systems Laboratory, School of Engineering and Applied Science, 1982.
- [27] E. Prange. The Use of Coset Equivalene in the Analysis and Decoding of Group Codes. Technical report, AIR FORCE CAMBRIDGE RESEARCH LABS HANSCOM AFB MA, 1959.
- [28] L. Rudolph. A class of majority logic decodable codes (corresp.). *IEEE Transactions on Information Theory*, 13(2):305–307, 1967.
- [29] B. Segre. Ovals in a finite projective plane. *Canadian Journal of Mathematics*, 7:414–416, 1955.
- [30] B. Segre. Lectures on modern geometry. *Bull. Amer. Math. Soc*, 67:442–443, 1961.

-
- [31] C. E. Shannon and W. Weaver. The mathematical theory of communication (urbana, il, 1949).
- [32] P. W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee, 1994.
- [33] P. Sin, J. Wu, and Q. Xiang. Dimensions of some binary codes arising from a conic in $pg(2, q)$. *Journal of Combinatorial Theory, Series A*, 118(3):853–878, 2011.
- [34] J. Singer. A theorem in finite projective geometry and some applications to number theory. *Transactions of the American Mathematical Society*, 43(3):377–385, 1938.
- [35] R. Singleton. Maximum distance q -nary codes. *IEEE Transactions on Information Theory*, 10(2):116–118, 1964.
- [36] R. M. Tanner. Minimum-distance bounds by graph analysis. *IEEE Transactions on Information Theory*, 47(2):808–821, 2001.
- [37] J.-P. Tillich. The decoding failure probability of mdpc codes. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 941–945. IEEE, 2018.
- [38] B. Vasic and O. Milenkovic. Combinatorial constructions of low-density parity-check codes for iterative decoding. *IEEE Transactions on information theory*, 50(6):1156–1176, 2004.
- [39] H. Zhang and J. M. Moura. Geometry based designs of LDPC codes. In *2004 IEEE International Conference on Communications (IEEE Cat. No. 04CH37577)*, volume 2, pages 762–766. IEEE, 2004.