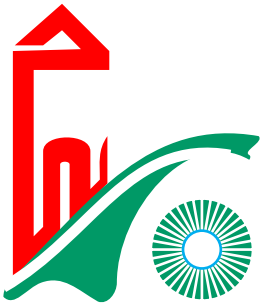
People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
Yahia Farès University of Médéa
Faculty of Sciences
Department of Mathematics and Computer Science



# Master's Dissertation

*Submitted in partial fulfillment of the requirements*

For the degree of master in mathematics

**Specialty :** Combinatorial Analysis, Probability and Applications

*by*

**Menasri Meriem**

## Code-based Cryptography

Publicly defended on 24.06.2025, before the jury composed of

| | | | |
|---|---|---|---|
| **President:** | Mrs. KOUIDER AIAD Soumia | *M.A.A.* | University of Médéa |
| **Examiner:** | Dr. AIT AMRANE N. Rosa | *M.C.B.* | University of Médéa |
| **Supervisor:** | Dr. BOUZARA Reguia Lamia | *M.C.B.* | University of Médéa |

**Academic Year: 2024-2025**

Code-based Cryptography

# *Acknowledgements*

First and foremost, I would like to thank **Allah**, the Merciful and Compassionate, for giving me the strength and courage to complete this modest work. Additionally, it is a pleasure to thank everyone who contributed to the completion of this dissertation.

First of all. I would like to express my sincerest thanks to my supervisor, Dr. **BOUZARA Reguia Lamia**, for guiding me throughout this study and for his scientific and moral support.

Mrs. **KOUIDER AIAD Soumia**, M.A.A. at Dr. Yahia Fares University of Medea, for honouring us by chairing the jury for this memory.

I would like to thank Dr. **AIT AMRANE N. Rosa**, M.C.B. at Dr.Yahia Fares the University of Medea, for honouring me by being a member of the jury and for dedicating some of his valuable time to read and correct this memory.

Finally. I would like to thank everyone who helped, whether directly or indirectly, in completing this project.

# إهداء

الحمد لله الذي أنعم وأكرم وأتم, مضى ماكان صعباً بلطف الله فما كنت لأفعل لولا أنك ياربي شددت عزمي ومكنتني وكنت العون لي

أهدي هذا العمل و أحتسب أجره كاملا الى من مرّت سنوات عمري بجانبها و كان قلبها لي في الحياة ركنا شديدا و من احاطتني بتراتيل دعواتها و ترقبت هذا النّجاح و لكن الأقدار شاءت أخذها

## يمّا -رحمها اللّه -

إلى من أخذ بيدي نحو كل نجاح،و فارق النوم لأجل راحتي وسعادتي، و من كانت دعواته تُمهّد لي الدرب وتُضيء لي الطريق **جدّي** ، حفظه الله

إلى أمّي و أفراد عائلتي و على رأسهم خوالي قاسمي : **محجوب , أيوب عبد القادر** كانو السّند الذي لا يميل و الشعور الصّادق الذي رافقني في هدوء و القوّة التي دفعتني لأصل لما أنا عليه اليوم

إلى من أدركت معها حبّ النجاح و التّطور و المسؤولية و كانت لي خير رفيق : الأستاذة **عايد وسيمة**

إلى صديقة اباهي برفقتها السنين و خير ما أهدتني جدران الجامعة عزيزة الروح : **عامر شيماء**

إلى التي لم تبخل عني بعلمها و توجيهاتها السديدة و كانت نعم العون و السند في إنجاز هذا العمل الدكتورة : **بوزارة رقية لامية**

﴿وَآخِرُ دَعْوَاهُمْ أَنِ الْحَمْدُ لِلَّهِ رَبِّ الْعَالَمِينَ﴾ 13 31

# Contents

**Acknowledgements** **i**

**Introduction** **2**

**1 Introduction to Cryptography** **3**

1.1 Types of Cryptographic Algorithms . . . . . 5

1.2 Secret Key Cryptography . . . . . 5

1.3 Public Key Cryptography . . . . . 7

**2 Finite Fields** **11**

2.1 Basic definitions: . . . . . 11

2.2 The prime subfield . . . . . 12

2.3 The ring $\mathbb{F}[X]$ . . . . . 13

2.4 Construction of finite fields . . . . . 14

**3 Introduction to Coding Theory: Error-correcting Codes** **19**

3.1 Basic Definitions . . . . . 20

3.2 Nearest Neighbor Decoding . . . . . 26

3.3 Syndrome Decoding . . . . . 27

3.4 Syndrome Decoding Procedure . . . . . 28

3.5 Goppa Codes . . . . . 30

3.6 Decoding Process . . . . . 36

**4 McEliece Public Key Encryption Cryptosystem** **39**

4.1 A Probabilistic Key Generation Algorithm . . . . . 40

4.2 Properties of the Encryption . . . . . 43

4.3 Decryption Algorithm . . . . . 43

4.4 Attacking . . . . . 46
4.5 Strengths and Weaknesses . . . . . 47

# Abstract

The first code-based cryptosystem was introduced in 1978 by McEliece, it is conjured to be post-quantum secure. In this thesis we present the McEliece cryptosystem, we start by introducing the fundamental terminology required for understanding the content that follows. Next, we delve into the definition and constraints of coding theory, as well as their role in general cryptographic systems. Finally, we focus specifically on the McEliece Cryptosystem, detailing how it employs Goppa codes and analyzing the security mechanisms that underpin this approach.

**Résumé**

Le premier système de la cryptographie à base de codes correcteurs d'erreurs a été introduit en 1978 par McEliece. Il est supposé être sécurisé face aux attaques quantiques. Dans ce mémoire, nous présentons le cryptosystème de McEliece. Nous commençons par introduire la terminologie fondamentale nécessaire à la compréhension du contenu qui suit. Ensuite, nous abordons les définitions et les contraintes de la théorie des codes, ainsi que leurs rôle dans les systèmes cryptographiques en général. Enfin, nous nous concentrons spécifiquement sur le cryptosystème de McEliece, en détaillant son utilisation des codes de Goppa et en analysant les mécanismes de sécurité qui sous-tendent cette approche .

# Notation

- $p$: Characteristic of the field (a prime number).
- $n$: Degree of the field extension.
- $q = p^n$: Total number of elements in the finite field.
- $\mathbb{Z}_n$: ring of integers modulo $n$.
- $\mathbb{Z}_{26}$: integers modulo 26, representing alphabet letters.
- $\mathbb{F}_p$: Prime field with $p$ elements.
- $\mathbb{F}_q$: Finite field with $q = p^n$ elements.
- $\mathbb{F}_q^\times$: Multiplicative group of non-zero elements in $\mathbb{F}_q$.
- $F, K$: Fields (usually $F$ is a finite field, $K$ is a subfield).
- $F[X], K[X]$: Polynomial rings over $F$ or $K$.
- $f(x)$: A polynomial in $F[X]$ or $K[X]$.
- $\deg(f)$: Degree of the polynomial $f(x)$.
- $\alpha \in F$: An element in the field $F$.
- $M(X)$: The minimal polynomial of $\alpha$ over $K$.
- $K(\alpha)$: Smallest field extension of $K$ containing $\alpha$.
- $\varphi(n)$: Euler's totient function.
- $[F : K]$: Degree of the field extension $F$ over $K$.
- $\cong$: Field isomorphism.

- $\langle f(x) \rangle$: Ideal generated by the polynomial $f(x)$ in $F[X]$.
- $F[X]/(f(x))$: Quotient ring; used to construct a finite field.

# Introduction

Cryptography relies on hard mathematical problems to transform data, ensuring the secure exchange of messages and protecting information from unauthorized modifications. Modern public-key cryptosystems are based on hard mathematical problems such as the discrete logarithm problem and integer factorization. So far, no known classical algorithm can efficiently solve these problems, making them a strong foundation for cryptographic security. However, with the advancement of quantum computing and the implementation of Shor's algorithm, these problems can be solved efficiently, posing a serious threat to existing cryptographic systems. Given that even a minor alteration of a single bit in digital communications can lead to catastrophic consequences. Cryptographic researchers are actively developing new cryptographic techniques based on mathematical problems that remain intractable even for quantum computers.

Recognizing the urgency of this issue, the National Institute of Standards and Technology (NIST) initiated a standardization process in 2016 to develop post-quantum cryptographic algorithms. The aim of this initiative is to establish cryptographic systems that are resilient against quantum attacks by relying on mathematically hard problems that remain secure even in the presence of quantum computers. One of the most promising candidates for post-quantum cryptography is Code-Based Cryptography (CBC), which leverages difficult problems in algebraic coding theory to provide quantum-resistant security.

The McEliece cryptosystem is a public-key encryption algorithm based on the coding theory, particularly Goppa codes. The security of the McEliece relies on general decoding problem which is NP hard and should resist the quantum-computers, making it a candidate for post-quantum cryptography.

In this thesis we present the McElice cryptosystem. It is organized as follows: In the first chapter we introduce cryptography, in the second chapter we give basic definitions of finite fields and the

construction of finite fields, in the third chapter we give a brief overview over linear codes and in the last chapter we present the McEliece cryptosystem. We conclude this work with a brief conclusion.

# Chapter 1

# Introduction to Cryptography

This chapter introduces the main types of cryptographic algorithms: symmetric-key, asymmetric-key, and hash functions. Each plays a key role in protecting data. For more details, you can refer to the sources listed in the references [5] [6] [7] [15] .
The science of secret writing, or cryptography, is an old skill. It was first applied in writing around 1900 B.C. When an Egyptian scribe employed unusual hieroglyphs in an inscription. With applications ranging from diplomatic messages to combat plans during times of conflict, some scholars argue that cryptography emerged on its own sometime after writing was invented. Therefore, it should come as no surprise that new cryptographic techniques emerged shortly after computer communications became widely used. When interacting over any untrusted media, which includes almost any network, especially the Internet, cryptography is required in data and telecommunications.

The Five Main Purposes of cryptography are:

- **Confidentiality/Privacy**: Ensuring only the intended recipient can read the message.
- **Authentication**: The process of verifying identity.
- **Integrity**: Proving the message hasn't been altered.
- **Non-repudiation**: Proving the sender actually sent the message.
- **Key Exchange**: The process of sharing cryptographic keys.

Furthermore, the next functions might be supported by cryptogrpahy:

- *Forward Secrecy*: This feature protects past encrypted sessions from compromise even if the server holding the messages is compromised. This is accomplished by creating a different key

for every session so that compromise of a single key does not threaten the entirely of the communications.

- *Optimal Security*: The system is characterised by its resilience to disruption, and the ciphertext is devoid of any information pertaining to the plaintext or the key. In order to achieve perfect security, it is essential that the key is at least as long as the plaintext, thereby rendering analysis and even brute-force attacks impossible. One-time pads serve as a prime example of such a system.

- *Deniable Authentication (also known as Message Repudiation)*: This is a method by which participants in an exchange of messages can be assured of the authenticity of the messages. Furthermore, it enables the senders to later plausibly deny their participation to a third party.

The unencrypted data is called **plaintext** and the encrypting plaintext is called **ciphertext** (it is typically decrypted back into plaintext). This procedure can be expressed as:

$$C = E_k(P).$$

$$P = D_k(C).$$

- $C$: is the ciphertext.
- $E$: is the encryption .
- $k$: is the key.
- $P$: is the plaintext.
- $D$: is the decryption technique.

It is evident that **cryptography** is most closely associated with the development and creation of the mathematical algorithms used to encrypt and decrypt messages. Conversely, **cryptanalysis** is the science of analysing and breaking encryption schemes. **Cryptology** is the overarching term denoting the comprehensive study of secret writing. It comprises both cryptography and cryptanalysis.

## 1.1 Types of Cryptographic Algorithms

Cryptographic algorithms can be categorized using the number of keys used for encryption and decryption as follows:

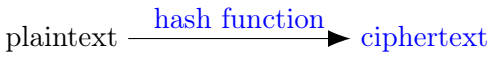
- **Secret Key Cryptography (SKC)**, commonly known as symmetric encryption, encrypts and decrypts data using a single key. Used primarily for confidentiality and privacy.




- **Asymmetric encryption**, commonly known as public key cryptography (PKC), uses one key for encryption and another for decryption. Mostly employed for key exchange, non-repudiation, and authentication.




- **Hash functions**: Permanently "encrypt" data via a mathematical process to create a digital fingerprint. Used mostly for the integrity of the message.
  Hash functions have no key since the plaintext is not recoverable from the ciphertext.




## 1.2 Secret Key Cryptography

Secret key cryptography algorithms use a single key for both encryption and decryption. The sender uses the key to encrypt the plaintext and transmits the ciphertext to the receiver. The receiver employs the identical key to decrypt the message and thereby recover the plaintext. Due to that fact the cryptographic process known as secret key cryptography is also referred as symmetric encryption.

It is evident that, in this instance, the key must be known to both the sender and the receiver,this is the secret. The most significant challenge associated with this approach pertains to the distribution of the key.

**Example 1** ***Simple substitution ciphers***

*As Julius Caesar surveys the unfolding battle from his hilltop outpost, an exhausted and disheveled courier bursts into his presence and hands him a sheet of parchment containing gibberish: j s j r d k f q q n s l g f h p g w j f p y m w t z l m n r r n s j s y q z h n z x Within moments, Julius sends an order for a reserve unit of charioteers to speed around the left flank and exploit a momentary gap in the opponent's formation. How did this string of seemingly random letters convey such important information? The trick is easy, once it is explained. Simply take each letter in the message and shift it five letters up the alphabet. Thus j in the ciphertext becomes e in the plaintext because e is followed in the alphabet by f,g,h,i,j. Applying this procedure to the entire ciphertext yields :*

| | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *J* | *S* | *J* | *R* | *D* | *K* | *F* | *Q* | *Q* | *N* | *S* | *L* | *G* | *F* | *H* | *P* | *G* | *W* | *J* | *F* | *P* | *Y* | *M* | *W* | *T* | *Z* | *X* |
| *E* | *N* | *E* | *M* | *Y* | *F* | *A* | *L* | *L* | *I* | *N* | *G* | *B* | *A* | *C* | *K* | *B* | *R* | *E* | *A* | *K* | *T* | *H* | *R* | *O* | *U* | *S* |

*The second line is the decrypted plaintext, and breaking it into words and supplying the appropriate punctuation, Julius reads the message*

*" Enemy falling back. Breakthrough imminent. Lucius "*

*There remains one minor quirk that must be addressed. What happens when Julius finds a letter such as d? There is no letter appearing five letters before d.*
*in the alphabet. The answer is that he must wrap around to the end of the alphabet. Thus d is replaced by y, since y is followed by z,a,b,c,d.*
*simple substitution cipher may be viewed as a rule or function*

$$\{a, b, c, d, e, ..., x, y, z\} \rightarrow \{A, B, C, D, E, ..., X, Y, Z\} \ .$$

***Statistical Attack****: The attack is based on frequency analysis. In lengthy texts, letters occur at different frequencies, which vary depending on the language.*

*To decrypt the plaintext message, the most frequently appearing symbol is substituted with the most common letter in the plaintext language, followed by the second most frequent symbol with the second most common letter, and so forth. This process is repeated until all symbols in the ciphertext have been replaced, revealing the original message. For example, the frequencies of the most commonly used letters in the French language, in order, are:*

| E | S | A | I | N | T | R | U | L | O | D |
|---|---|---|---|---|---|---|---|---|---|---|
| 14.69% | 8.01% | 7.54% | 7.18% | 6.89% | 6.88% | 6.49% | 6.12% | 5.63% | 5.29% | 3.66% |

**Definition .1** • *$\mathbb{Z}_{26}$: the set of integers modulo 26, i.e. $\{0, 1, 2, \ldots, 25\}$, with addition and multiplication modulo 26. Commonly used to represent the 26 letters of the English alphabet in classical cryptography.*

**Example 2** ***Affine Cipher***

*An encipherment scheme (or algorithm) of the form*

$$E(a, b) : \mathbb{Z}_{26} \to \mathbb{Z}_{26},$$

$$x \mapsto ax + b \mod 26.$$

*Is called an affine cipher. Here $x$ is the numerical equivalent of the given plaintext letter,and $a$ and $b$ are (appropriately chosen) integers,*
*If $y = E(x) = (ax + b) \mod 26$, then we can "solve for $x$" in terms of $y$ and so determine $E^{-1}(y)$. That is, if $y \equiv (ax + b) \mod 26$, then*

$$y - b \equiv ax \pmod{26}, \quad \text{or equivalently} \quad ax \equiv (y - b) \mod 26.$$

*Using our earlier results, we see that if we multiply both sides by $a^{-1} \mod 26$, then*

$$x \equiv a^{-1}(y - b) \mod 26$$

*and so our decipherment function is:*

$$E^{-1}(y) = a^{-1}(y - b) \mod 26.$$

## 1.3 Public Key Cryptography

Public key cryptography represents the most significant new development in cryptography for the past 300-400 years. The first documented public exposition of modern PKC was by Martin Hellman, a professor at Stanford University, and Whitfield Diffie, a graduate student, in 1976. The aforementioned paper described a two-key cryptosystem in which two parties could engage in secure communication over a non-secure communications channel without having to share a secret key.

While the above examples may be considered trivial, they do represent two of the functional pairs that are used with PKC. Specifically, these pairs are characterized by the ease of multiplication and exponentiation, in contrast to the relative difficulty of factoring and calculating logarithms, respectively. The mathematical "trick" in PKC is to find a trap door in the one-way function, thereby facilitating the inverse calculation given knowledge of some item of information.

Generic PKC employs two keys that are mathematically related, yet knowledge of one key does not allow for the straightforward determination of the other. The encryption of the plaintext is accomplished by the utilisation of a single key, while the decryption of the ciphertext is facilitated by the employment of a second key. The salient point is that the order in which keys are applied is inconsequential, both keys are requisite for the process to function. The necessity of a pair of keys is the reason for the designation of this approach as asymmetric cryptography.

In PKC, one of the keys is designated the public key and the owner is at liberty to advertise it as widely as they wish. The other key is designated the private key and is never revealed to another party. The transmission of messages is a straightforward process within the framework of this scheme. In the following hypothesis, it is assumed that Alice wishes to transmit a message to Bob. In this scenario, Alice encrypts a set of information using Bob's public key, and Bob, in turn, decrypts the ciphertext using his private key. This method could also be used to prove who sent a message, for example, Alice could encrypt some plaintext with her private key, and when Bob decrypts using Alice's public key, he knows that Alice sent the message (authentication) and Alice cannot deny having sent the message (non-repudiation). The following public key cryptography algorithms are currently employed for key exchange or digital signatures:

- **RSA**: The first, and still most common, PKC implementation is named for the three MIT mathematicians who developed it Ronald Rivest, Adi Shamir, and Leonard Adleman. RSA is utilised in a multitude of software products in the present day, with its application encompassing key exchange, digital signatures and the encryption of diminutive data blocks. It is evident that RSA employs a variable-size encryption block and a variable-size key. The key-pair is derived from a very large number, n, which is the product of two prime numbers chosen according to special rules, these primes may be 100 or more digits in length each, yielding an n with roughly twice as many digits as the prime factors. The public key information includes n and a derivative of one of the factors of n. It is therefore impossible for an attacker to determine the prime factors of n from this information alone, and this is what makes the RSA algorithm so secure.

  The ability for computers to factor large numbers, and therefore attack schemes such as RSA,

is rapidly improving, and systems today can find the prime factors of numbers with more than 200 digits.

However, if a large number is created from two prime factors of a similar size, there is no known factorisation algorithm that will solve the problem in a reasonable amount of time. For example, a 2005 test to factor a 200-digit number took 1.5 years and over 50 years of compute time. In 2009, Kleinjung et al. Reported that factoring a 768-bit (232-digit) RSA-768 modulus utilising hundreds of systems took two years (Kleinjung et al., 2009). They further estimated that factoring a 1024-bit RSA modulus would take approximately one thousand times as long (ibid.). Nevertheless, the recommendation was made that 1024-bit RSA be phased out by 2013. Nevertheless, one might presume that the RSA algorithm offers protection in the form of its ability to allow users to easily increase the key size, thus ensuring that they can maintain a position ahead of the computer processing curve. As a tangential observation, the patent for RSA expired in September 2000. This did not appear to have any impact on RSA's popularity.

- **Diffie-Hellman:** Following the publication of the RSA algorithm, Diffie and Hellman developed their own algorithm. Diffie-Hellman is exclusively employed for the purpose of secret-key key exchange, it is not used for authentication or digital signatures.

- **Digital Signature Algorithm (DSA)**: The algorithm specified in the **Digital Signature Standard (DSS)** published by the National Institute of Standards and Technology (NIST) is designed to facilitate the process of digital signature creation, with the purpose of message authentication. The subject has been delineated in FIPS PUB 186-4.

- **ElGamal:** The ElGamal PKC system was designed by Taher Elgamal. It is similar to the Diffie-Hellman system and is used for key exchange. ElGamal is used in certain subsequent iterations of Pretty Good Privacy (PGP), in addition to GNU Privacy Guard (GPG), along with other cryptosystems.

- **Elliptic Curve Cryptography (ECC)**: A PKC algorithm based on elliptic curves. ECC can offer levels of security with small keys comparable to those of RSA and other PKC methods. The system was designed for devices with limited computing power and/or memory, such as smartcards and PDAs.

- **Identity-Based Encryption (IBE)**: IBE is a novel scheme that was first proposed by Adi Shamir in 1984. The PKC-based key authentication system under discussion is such

that the public key of a user may be derived from unique information pertaining to that user's identity. This system thus allows two users to exchange encrypted messages without the need for a priori knowledge of each other. In 2001, Dan Boneh (Stanford) and Matt Franklin (U.C. Davis) developed a practical implementation of IBE based on elliptic curves and a mathematical construct termed the Weil Pairing. In that same year, Clifford Cocks of GCHQ also described another IBE solution based on quadratic residues in composite groups. RFC 5091: Identity-Based Cryptography Standard (IBCS) provides a detailed description of an implementation of IBE (Identity-Based Encryption) using Boneh-Franklin (BF) and Boneh-Boyen (BB1) Identity-Based Encryption algorithms.

# Chapter 2

# Finite Fields

In this chapter, we discuss the fundamental concepts of finite fields, which are essential in many areas of mathematics and coding theory. We begin with basic definitions and then examine the prime subfield and the polynomial ring $\mathbb{F}[X]$. Finally, we explore how finite fields can be constructed. The content presented is supported by references such as [1] [2] [14].

## 2.1 Basic definitions:

We start by giving some basic definitions over finite fields.

**Definition .2** *A field is defined as a set $\mathbb{F}$ equipped with two operations, addition $(+)$ and multiplication $(\cdot)$, along with two distinguished elements 0 and 1, such that the following conditions are satisfied [2]:*

- $(\mathbb{F}, +)$ *forms an abelian group with identity element 0.*
- $(\mathbb{F}^*, \cdot)$ *where $\mathbb{F}^* = \mathbb{F} \setminus \{0\}$, is an abelian group with identity element 1.*
- *For every $a \in \mathbb{F}$, we have that that $a \cdot 0 = 0$.*
- *For all $a, b, c \in \mathbb{F}$, we have that $a \cdot (b + c) = a \cdot b + a \cdot c$.*

**Remark 1** - *A finite field is simply a field with a finite number of elements.*

- *The field of $q$ elements is denoted by $\mathbb{F}_q$ .*

The most basic example of a finite field is $\mathbb{F}_p$ where $p$ is a prime number.

**Definition .3** *The field $\mathbb{F}_p$ is defined as the quotient ring $\mathbb{Z}/p\mathbb{Z}$, such that the elements of $\mathbb{F}_p$ are the integers $\{0, 1, \ldots, p-1\}$, and both addition and multiplication are carried out modulo $p$.*

To confirm that this structure is indeed a field, the crucial step is to show that every non-zero element $a \in \mathbb{F}_p$ has a multiplicative inverse. Since $p$ is prime, any such $a$ is coprime with $p$, meaning $\gcd(a, p) = 1$. By Euclid's algorithm, there exist integers $x$ and $y$ such that $ax + py = 1$, which implies $ax \equiv 1 \mod p$. Therefore, $x$ serves as the multiplicative inverse of $a$ in $\mathbb{F}_p$[2].

## 2.2 The prime subfield

Let $\mathbb{F}$ be a finite field. For a positive integer $r$, consider the repeated sum $s_r = 1+1+\cdots+1$ (with $r$ terms). Since $\mathbb{F}$ contains finitely many elements, there must exist some $r$ for which $s_r = 0$.

Define $p$ to be the smallest positive integer such that $s_p = 0$. To see that such $p$ must be prime, suppose otherwise: if $p = a \cdot b$ with $a, b < p$, then using the distributive property, we would get $0 = s_p = s_a \cdot s_b$, implying that either $s_a = 0$ or $s_b = 0$, which contradicts the minimality of $p$. Hence, $p$ must be prime.

**Definition .4** *The value $p$ is called the **characteristic** of the field $\mathbb{F}$[1].*

**Remark 2** *The subset $\{0, s_1, s_2, \ldots, s_{p-1}\} \subseteq \mathbb{F}$ forms a subfield of $\mathbb{F}$, and it is isomorphic to $\mathbb{F}_p$.*

**Definition .5** *The subset $\{0, s_1, s_2, \ldots, s_{p-1}\} \subseteq \mathbb{F}$ is called **prime subfield** of $\mathbb{F}$[1].*

The essential idea for classifying all finite fields lies in the fact that $\mathbb{F}$ can be viewed as a finite-dimensional vector space over $\mathbb{F}_p$. Let $n = \dim_{\mathbb{F}_p} \mathbb{F}$. Then it follows that $|\mathbb{F}| = p^n$. In particular, the number of elements in any finite field must be a power of a prime.

## 2.3 The ring $\mathbb{F}[X]$

**Definition .6** *Let $\mathbb{F}$ be a field, the set of polynomials with coefficients in $\mathbb{F}$ is given by:*

$$\mathbb{F}[x] := \left\{ \sum_{i=0}^{n} a_i x^i \;\middle|\; a_i \in \mathbb{F},\ n \geq 0 \right\}.$$

*Equipped with the usual addition and multiplication of polynomials, forms a ring known as the polynomial ring over $\mathbb{F}$.*

*Each element of $\mathbb{F}[x]$ is called a polynomial over $\mathbb{F}$, and the degree of a polynomial is the largest exponent of $x$ that appears with a non-zero coefficient.*

*A polynomial $f(x)$ of degree greater than zero is called* reducible over $\mathbb{F}$ *if it can be factored into two non-constant polynomials $g(x), h(x) \in \mathbb{F}[x]$ such that $f(x) = g(x)h(x)$ and $\deg g(x), \deg h(x) < \deg f(x)$. If no such factorization exists, $f(x)$ is said to be* irreducible *over $\mathbb{F}$[14].*

**Properties 1**

- *$\mathbb{F}[X]$ is a principal ideal domain: every ideal in $\mathbb{F}[X]$ is generated by a single polynomial $g(X) \in \mathbb{F}[X]$.*
- *$\mathbb{F}[X]$ is a unique factorization domain.*
- *By the Remainder Theorem: $P(X) \in \mathbb{F}[X]$ vanishes at a point $\alpha \in \mathbb{F}$ if and only if $(X - \alpha)$ divides $P(X)$.*

**Example 3** *$\mathbb{Z}/\mathbb{Z}_3[X]$ is the set of polynomials with coefficient in $\mathbb{Z}_3 = \{0, 1, 2\}$. Let $A = 1 + 2X$ and $B = 2 + X^2$, then:*
***Addition:*** $A + B = (1 + 2) + 2X + X^2 = 3 + 2X + 1X^2 = 2X + X^2$ .
***Multiplication:*** $AB = (1 + 2X)(2 + X^2) = 2 + 4X + X^2 + 2X^3 = 2 + X + X^2 + 2X^3$.

**Theorem .1** *If $P(X) \in \mathbb{F}[X]$ is non-zero and has degree at most $d$, then it has at most $d$ roots in $\mathbb{F}$[14].*

We also define the derivative map $D : \mathbb{F}[X] \to \mathbb{F}[X]$, by setting $D(X^i) = iX^{i-1}$, and extending it linearly. This map satisfies the product rule:

$$D(fg) = f \cdot D(g) + D(f) \cdot g.$$

We will often write $f'(X)$ for $D(f(X))$[14].

**Remarks 1**

- *Although derivatives do not have the same geometric meaning as over* $\mathbb{R}$*, they are still quite useful. In particular,* $(X-\alpha)^2$ *divides* $P(X)$ *if and only if* $P(\alpha)=0$ *and* $P'(\alpha)=0$*.*
- *For detecting higher multiplicity roots (e.g., triple roots), higher-order derivatives* $D^k$ *may fail, especially in fields of small characteristic. For instance, in characteristic 2,* $D^2(f(X))=0$ *for all* $f(X)$*.*

## 2.4 Construction of finite fields

### 2.4.1 The field $\mathbb{F}_p/\langle f(x)\rangle$:

**Theorem .2** *Let* $f(x)\in\mathbb{F}_p[x]$ *be a monic irreducible polynomial of degree* $n$*, the ring* $\mathbb{F}_p[x]/f(x)$ *is a field of order* $n$ *[14].*

**proof.**

The cosets mod $f(x)$ are represented by remainders
$a_0+a_1x+\cdots+a_{n-1}x^{n-1}$ ; $a_i\in\mathbb{F}_p$, thus there are $p^n$ element of these. Since $f(x)$ is irreducible, then $\mathbb{F}_p[x]/f(x)$ is a field. ∎

**Example 4** *The first example is the construction of the field.* $\mathbb{F}_4$*:*
*We have that the polynomial* $f(x)=X^2+X+1$ *is irreducible over* $\mathbb{F}_2[X]$*, then using this polynomial we can construct a field as follows:* $\mathbb{F}_2/(X^2+X+1)$ *such that* $\mathbb{F}_4=\{0,1,X,X+1\}$ *with* $X^2=X+1$ *and* $1=-1$*.*
*The additive law of the field is given by:*

| $+$ | $1$ | $X$ | $X+1$ |
|---|---|---|---|
| $1$ | $0$ | $X+1$ | $X$ |
| $X$ | $X+1$ | $0$ | $1$ |
| $X+1$ | $X$ | $1$ | $0$ |

*The multiplicative law of the field is given by:*

| $\times$ | $X$ | $X+1$ |
|---|---|---|
| $X$ | $X+1$ | $1$ |
| $X+1$ | $1$ | $X$ |

**Example 5** *The second example is the construction of the field* $\mathbb{F}_8$*:*

*Let* $g(x) = X^3 + X + 1$ *be a polynomial over* $\mathbb{F}_2$*,* $g(x)$ *is irreducible because it has no roots in* $\mathbb{F}_2$ *then the quotient* $\mathbb{F}_8 = \mathbb{F}_2/\langle g(X)\rangle$ *is a field. To simplify the multiplication we will use the next tabular:*

| $\times$ | $X$ | $X^2$ |
|---|---|---|
| $X$ | $X^2$ | $X+1$ |
| $X^2$ | $X+1$ | $X$ |

*The elements of* $\mathbb{F}_8$ *are* $0, 1, X, X^2, X+1, X+X^2, 1+X^2, 1+X+X^2$ *and the multiplicative law is given as follows:*

| | | | | | | |
|---|---|---|---|---|---|---|
| $1$ | $X$ | $X^2$ | $X+1$ | $X^2+X$ | $X^2+X+1$ | $X^2+1$ |
| $X$ | $X^2$ | $X+1$ | $X^2+X$ | $X^2+X+1$ | $X^2+1$ | $1$ |
| $X^2$ | $X+1$ | $X^2+X$ | $X^2+X+1$ | $X^2+1$ | $1$ | $X$ |
| $X+1$ | $X^2+X$ | $X^2+X+1$ | $X^2+1$ | $1$ | $X$ | $X^2$ |
| $X^2+X$ | $X^2+X+1$ | $X^2+1$ | $1$ | $X$ | $X^2$ | $X+1$ |
| $X^2+X+1$ | $X^2+1$ | $1$ | $X$ | $X^2$ | $X+1$ | $X^2+X$ |
| $X^2+1$ | $1$ | $X$ | $X^2$ | $X+1$ | $X^2+X$ | $X^2+X+1$ |

**Theorem .3** *Every finite field has prime power order* $p^n$*[1].*

**Theorem .4** *If* $\mathbb{F}$ *is a finite field, the group* $\mathbb{F}^\times$ *is cyclic[1].*

**Lemma 1** *The multiplicative order of any non-zero* $\alpha \in \mathbb{F}_q$ *divides* $q-1$*. Moreover, if* $\alpha, \beta \in \mathbb{F}_q^*$ *are such that* $\gcd(|\alpha|, |\beta|) = 1$*, then* $|\alpha\beta| = |\alpha| \cdot |\beta|$*.*

**Theorem .5** *A non-zero element of* $\mathbb{F}_q$ *is a primitive element if and only if its multiplicative order is* $q-1$*. Every finite field has at least one primitive element[14].*

Primitive elements are not unique in a given finite field. For any $\alpha \in \mathbb{F}_{q^n}$, a *minimal polynomial* over $\mathbb{F}_q$ is the unique monic polynomial $f(x) \in \mathbb{F}_q[x]$ of lowest degree such that $f(\alpha) = 0$. This polynomial is always irreducible and its roots in $\mathbb{F}_{q^n}$ are all primitive elements.

**Definition .7** *(Primitive Polynomial) An irreducible polynomial* $p(x)$ *of degree* $m$ *over* $\mathbb{F}_q$ *is called primitive polynomial if its roots form primitive elements of* $\mathbb{F}_{q^m}$*[1].*

*For example,*

$$p(x) = x^{13} + x^5 + x^3 + x + 1,$$

*is a primitive polynomial of degree 13 over binary field* $\mathbb{F}_2$.

**Remark 3** *There are* $\frac{\phi(q^n-1)}{n}$ *primitive polynomials of degree* $n$ *over* $\mathbb{F}_q$.

### 2.4.2 Constructing algebraic extensions

Let $\mathbb{F}$ be a field and let $K$ be a subfield. Note that $\mathbb{F}$ is a vector space over $K$. We denote $\dim_K(\mathbb{F})$ by $[\mathbb{F} : K]$. Suppose $[\mathbb{F} : K] = n$. Now pick any $\alpha \in \mathbb{F}$. Consider the elements $1, \alpha, \alpha^2, \ldots, \alpha^n$. Since $\mathbb{F}$ is $n$-dimensional over $K$, these $n+1$ elements must be linearly dependent over $K$. Thus $\alpha$ is the root of some non-zero $P(X) \in K[X]$. This means every $\alpha \in \mathbb{F}$ is *algebraic* over $K$.

**Theorem .6** *Every finite field* $\mathbb{F}$ *of characteristic* $p$ *is a finite algebraic extension field of* $\mathbb{F}_p$*[1].*

Conversely, every finite algebraic extension field of $\mathbb{F}_p$ is a finite field. This gives us a characterization of all finite fields.

Let $\mathbb{F}$ be a field and let $\mathbb{K}$ be a subfield with $[\mathbb{F} : \mathbb{K}]$ is finite .

Now consider an element $\alpha \in \mathbb{F}$. Let $M(X) \in \mathbb{K}[X]$ be a non-zero monic polynomial of smallest degree such that $M(\alpha) = 0$. Note that $M(X)$ must be irreducible in $\mathbb{K}[X]$. Also note that if $P(X) \in \mathbb{K}[X]$ is such that $P(\alpha) = 0$, then $M(X) \mid P(X)$. In particular, $M(X)$ is the unique nonzero monic polynomial of smallest degree for which $M(\alpha) = 0$. This $M(X)$ is called the minimal polynomial of $\alpha$ over $\mathbb{K}$.

**Definition .8** $\mathbb{K}(\alpha)$ *is the smallest subfield of* $\mathbb{F}$ *containing both* $\mathbb{K}$ *and* $\alpha$.
*Observe that if* $d = \deg(M(X))$*, then*

$$\mathbb{K}(\alpha) = span_{\mathbb{K}}\{1, \alpha, \alpha^2, \ldots, \alpha^{d-1}\}.$$

*This equality holds between the two subsets of* $\mathbb{F}$*, in particular it assumes that we already have our hands on* $\mathbb{F}$*, and describes* $\mathbb{K}(\alpha)$ *in terms of the operations of* $\mathbb{F}$.

The key to getting a "construction" of $\mathbb{F}$ using only operations from $\mathbb{K}$ is the following field isomorphism:

$$\mathbb{K}(\alpha) \cong \mathbb{K}[X]/(M(X)).$$

(This is the ring $\mathbb{K}[X]$ mod the ideal generated by $M(X)$). The right-hand side is defined only in terms of $\mathbb{K}$, and is called a primitive field extension. If $d = \deg(M(X))$, then the extension is of degree $d$. This gives us the following construction of $\mathbb{F}$ from $\mathbb{K}$. Set $K_0 = \mathbb{K}$. Let $\alpha_1 \in \mathbb{F} \setminus K_0$, and set $K_1 = K_0(\alpha_1)$. If $K_1 = \mathbb{F}$, then we are done. Otherwise, pick $\alpha_2 \in \mathbb{F} \setminus K_1$, set $K_2 = K_1(\alpha_2)$, etc. At each stage, $[\mathbb{F} : K_i]$ reduces, and thus the process must stop. This gives us a construction of $\mathbb{F}$ as a finite sequence of primitive extensions of $\mathbb{K}$. It can be completely specified by the sequence of irreducible polynomials $M_i(X) \in K_i[X]$, where $M_i(X)$ is the minimal polynomial of $\alpha_{i+1}$ over $K_i$[14].

**Example 6** *Let* $\mathbb{F}_{2^4}$ *be the field isomorphic to* $\frac{\mathbb{F}_2[x]}{\langle x^4+x+1\rangle}$. *Let* $\alpha$ *be a root of* $x^4 + x + 1$, *since multiplicative order of* $\alpha$ *is 15, it can be used to generate all elements of* $\mathbb{F}_{2^4}$, *equivalently,* $\mathbb{F}_{2^4} = \langle\alpha\rangle$. *Hence we can represent elements of* $\mathbb{F}_{2^4}$ *as:*

$$\begin{aligned}
0 &= 0,\\
1 &= 1,\\
\alpha &= \alpha,\\
\alpha^2 &= \alpha^2,\\
\alpha^3 &= \alpha^3,\\
\alpha^4 &= 1+\alpha,\\
\alpha^5 &= \alpha+\alpha^2,\\
\alpha^6 &= \alpha^2+\alpha^3,\\
\alpha^7 &= 1+\alpha+\alpha^3,\\
\alpha^8 &= 1+\alpha^2,\\
\alpha^9 &= \alpha+\alpha^3,\\
\alpha^{10} &= 1+\alpha+\alpha^2,\\
\alpha^{11} &= \alpha+\alpha^2+\alpha^3,\\
\alpha^{12} &= 1+\alpha+\alpha^2+\alpha^3,\\
\alpha^{13} &= 1+\alpha^2+\alpha^3,\\
\alpha^{14} &= 1+\alpha^3.
\end{aligned}$$

**Definition .9** *Let* $n$ *be a natural number. The space* $\mathbb{F}_q^n$ *is defined as:*

$$\mathbb{F}_q^n = \{(a_1, a_2, \ldots, a_n) \mid a_i \in \mathbb{F}_q \text{ for } 1 \leq i \leq n\}.$$

*With addition defined coordinate-wise. This structure is called an* $n$*-dimensional vector space over*

$\mathbb{F}_q$[14].

*Let $V$ be a non-empty set. Then $V$ is a* ***vector space over*** *$\mathbb{F}_q$ if for all $\mathbf{u}, \mathbf{v}, \mathbf{w} \in V$ and all $\lambda, \mu \in \mathbb{F}_q$, the following properties hold:*

- $\mathbf{u} + \mathbf{v} \in V$.
- $(\mathbf{u} + \mathbf{v}) + \mathbf{w} = \mathbf{u} + (\mathbf{v} + \mathbf{w})$.
- *There exists* $\mathbf{0} \in V$ *such that* $\mathbf{0} + \mathbf{v} = \mathbf{v}$.
- *For each* $\mathbf{v} \in V$*, there exists* $-\mathbf{v} \in V$ *such that* $\mathbf{v} + (-\mathbf{v}) = \mathbf{0}$.
- $\mathbf{u} + \mathbf{v} = \mathbf{v} + \mathbf{u}$.
- $\lambda\mathbf{v} \in V$.
- $\lambda(\mathbf{u} + \mathbf{v}) = \lambda\mathbf{u} + \lambda\mathbf{v}$.
- $(\lambda + \mu)\mathbf{v} = \lambda\mathbf{v} + \mu\mathbf{v}$.
- $(\lambda\mu)\mathbf{v} = \lambda(\mu\mathbf{v})$.
- $1 \cdot \mathbf{u} = \mathbf{u}$ *if* $1$ *is the multiplicative identity of* $\mathbb{F}_q$.

# Chapter 3

# Introduction to Coding Theory: Error-correcting Codes

*This chapter explores how error-correcting codes are used to detect and fix errors in data transmission. It introduces basic concepts, explains syndrome decoding and its procedure, and presents Goppa codes with their decoding process. These tools are essential for building reliable communication systems The content presented is supported by references such as [10] [14] .*

*The purpose of coding theory is to ensure reliable communication over noisy channels. This process involves encoding, transmission through a potentially noisy channel, and decoding. The basic model is illustrated below:*

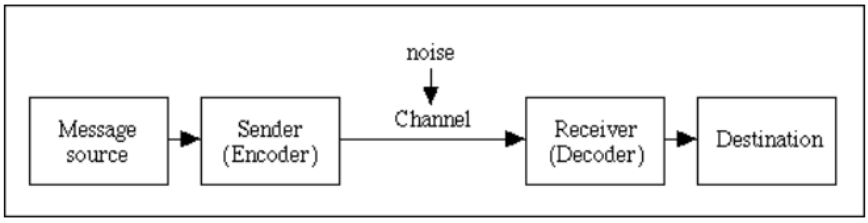



***Figure 1:*** *A conceptual view of message transmission over a noisy channel.*

*Suppose a message $m$ is sent and $e$ is the error introduced during transmission. If $y$ is the received message, then:*

$$y = m + e.$$

*Where '+' denotes coordinate-wise addition.*

## 3.1 Basic Definitions

**Definition .10** *A* block code $\mathcal{C}$ *of length* $n$ *is a subset of* $\mathcal{A}^n$*, where* $\mathcal{A}$ *is the alphabet set. Typically,* $\mathcal{A} = \mathbb{F}_q$*[14].*

**Remark 4**

- *If every codeword consists of repetitions of a single symbol, the code is called a* repetition code.

**Definition .11** *The* Hamming distance $d_H(x, y)$ *between two vectors* $x, y \in \mathbb{F}_q^n$ *is the number of positions where* $x$ *and* $y$ *differ [14].*

**Definition .12** *The* Hamming weight $w(x)$ *of* $x \in \mathbb{F}_q^n$ *is the number of non-zero components of* $x$ *[14].*

**Definition .13** *The* minimum Hamming distance *of a code* $\mathcal{C}$ *is defined by [14]:*

$$d(\mathcal{C}) := \min\{d_H(x, y) \mid x, y \in \mathcal{C},\ x \neq y\}.$$

**Remark 5** *For any* $x \in \mathbb{F}_q$*, the* ***Hamming weight*** *is defined as:*

$$w(x) = d(x, 0) = \begin{cases} 1 & \text{if } x \neq 0, \\ 0 & \text{if } x = 0. \end{cases}$$

**Theorem .7 (Hamming Distance)** *The Hamming distance defines a metric on* $\mathbb{F}_q^n$*. For* $x, y \in \mathbb{F}_q^n$*, we have:*

$$d_H(x, y) = w(x - y).$$

*In particular,* $d(x, y) = w(x+y)$ *when the field is of characteristic 2. Furthermore, for any* $x, y \in \mathbb{F}_2^n$*,*

$$w(x + y) = w(x) + w(y) - 2w(x \star y).$$

*Where* $\star$ *denotes component-wise multiplication [14].*

**Definition .14** ***Encoding*** *is the process:*

$$\textit{Encode:}\{\textit{Messages set}\} \longrightarrow \textit{Code } \mathcal{C} \subseteq \mathbb{F}_q^n.$$

*Encoding may not always be straightforward. For example, suppose we want to encode two messages: "Yes" and "No". We may map "Yes"* $\to 1$ *and "No"* $\to 0$*, but due to noise, a transmitted 1 might be received as 0, making decoding ambiguous.*

*To address this, one can use repetition-based schemes. For example:*

$$\textit{Yes} \to 111, \quad \textit{No} \to 000.$$

*A single error can be corrected via majority decoding.*

*In binary repetition code of length 3 over* $\mathbb{F}_2$*, the majority bit decoding method is used. This leads to the essential challenge of coding theory:* ***Error Detection and Correction*** *[14].*

**Theorem .8** *A code* $\mathcal{C}$ *can detect up to* $s$ *errors if* $d(\mathcal{C}) \geq s+1$*, and it can correct up to* $t$ *errors if* $d(\mathcal{C}) \geq 2t+1$ *[14].*

**Definition .15 (Sphere)** *A* sphere *of radius* $r$ *centered at* $u \in \mathbb{F}_q^n$ *is defined as:*

$$S(u,r) := \{v \in \mathbb{F}_q^n \mid d(u,v) \leq r\}.$$

**Theorem .9** *Let* $u \in \mathbb{F}_q^n$ *and* $r \in \{0,1,\ldots,n\}$*, then the number of elements in* $S(u,r)$ *is:*

$$\sum_{i=0}^{r} \binom{n}{i} (q-1)^i.$$

**Definition .16 (Perfect Code)** *A code is said to be* perfect *if it achieves equality in the sphere packing bound [14].*

**Example 7** *Examples of perfect codes include:*

- *The binary repetition code of length 5.*
- *The code containing all elements of* $\mathbb{F}_q^n$*.*

### 3.1.1 Linear Codes

*Linear codes of length $n$ over $\mathbb{F}_q$ are subspaces of $\mathbb{F}_q^n$, so their dimensions are finite. Since a linear code is a vector space, all of its elements can be described using a basis.*

**Definition .17** *Let $\mathcal{C}$ be a $k$-dimensional subspace of $\mathbb{F}_q^n$. Then $\mathcal{C}$ is called a $q$-ary $[n, k]$ linear code. If $d(\mathcal{C}) = d$, then $\mathcal{C}$ is a $q$-ary $[n, k, d]$ code over $\mathbb{F}_q$ [14].*

**Example 8** ***Linear codes***

- $C_1 = \{00, 01, 10, 11\}$ *is a* $[2, 2, 1]$*-code.*
- $C_2 = \{000, 011, 101, 110\}$ *is a* $[3, 2, 2]$*-code.*
- $C_3 = \{00000, 01101, 10110, 11011\}$ *is a* $[5, 2, 3]$*-code.*

**Lemma 2** *A $q$-ary $[n, k, d]$ code contains $q^k$ codewords. For $x, y \in \mathbb{F}_q^n$, we have $d(x, y) = w(x - y)$.*

**Theorem .10** *Let $\mathcal{C}$ be a linear code. Then $d(\mathcal{C})$ equals the minimum weight among all non-zero codewords of $\mathcal{C}$:*

$$d(\mathcal{C}) = w(\mathcal{C}).$$

**Definition .18** *The generator matrix $G$ of a $q$-ary linear $[n, k]$ code $\mathcal{C}$ is a $k \times n$ matrix whose rows form a basis of the vector space $\mathcal{C}$ over $\mathbb{F}_q$ [14].*

**Example 9** ***Generator matrix***

*The generator matrix of the $[3, 2, 2]$-code $C = \{000, 101, 011, 110\}$ is given by:*

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

**Definition .19** *Let $\mathcal{C}$ be a $q$-ary linear $[n, k]$ code with generator matrix $G$. Then a $(n - k) \times n$ matrix $H$ is called a parity-check matrix for $\mathcal{C}$ if:*

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

*If $\mathcal{C}$ is an $[n, k]$ linear code, then the generator matrix is $k \times n$, and the parity-check matrix is $(n - k) \times n$.*

*The number of generator matrices for a linear code often exceeds one due to multiple choices of basis and permutations of rows. Even if the basis is fixed, row permutations result in different generator matrices[14].*

**Remark 6** *The rows of $G$ are linearly independent.*

**Example 10** *The code $\mathcal{C} = \{000, 001, 100, 101\}$ is a 2-dimensional code, and its generator matrix does not have a standard form.*

**Lemma 3** *Let $\mathcal{C}$ be an $[n, k]$–linear code over $\mathbb{F}_q$ with generator matrix $G$. Then $v \in \mathbb{F}_q^n$ belongs to $\mathcal{C}^\perp$ if and only if $v$ is orthogonal to every row of $G$, i.e., $v \in \mathcal{C}^\perp \iff vG^T = 0$.*

*In particular, given an $(n-k) \times n$ matrix $H$, then $H$ is a parity-check matrix for $\mathcal{C}$ if and only if the rows of $H$ are linearly independent and $HG^T = 0$.*

**Theorem .11** *Let $\mathcal{C}$ be a linear code and let $H$ be a parity-check matrix for $\mathcal{C}$. Then:*

*(i) $\mathcal{C}$ has distance $\geq d$ if and only if any $d-1$ columns of $H$ are linearly independent.*

*(ii) $\mathcal{C}$ has distance $\leq d$ if and only if $H$ has $d$ columns that are linearly dependent.*

**Corollary 1** *Let $\mathcal{C}$ be a linear code and let $H$ be a parity-check matrix for $\mathcal{C}$. Then the following statements are equivalent:*

*(i) $\mathcal{C}$ has distance $d$;*

*(ii) Any $d-1$ columns of $H$ are linearly independent and $H$ has $d$ columns that are linearly dependent.*

**Example 11** ***Parity-check matrix of a binary Hamming code:***

- *The parity-check matrix $H$ of a binary $[7, 4]$ Hamming code is given by*

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

- *The minimum Hamming distance of a binary Hamming code is $d = 3$ (The first three columns are linearly dependent).*

- *Let* $v = (0, 0, 1, 1, 0, 0, 1)$ *be a code word then:*

$$Hv^t = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

**Definition .20 (Dual Code)** *For a linear code* $C \subseteq \mathbb{F}_q^n$*, its* dual code *is:*

$$C^\perp := \{\mathbf{v} \in \mathbb{F}_q^n \mid \mathbf{u} \cdot \mathbf{v} = 0 \text{ for all } \mathbf{u} \in C\}.$$

*Where* $\cdot$ *denotes the standard Euclidean dot product.*

**Theorem .12 (Properties of Dual Codes)** *[14] For any* $[n, k]$*-linear code* $C$ *over* $\mathbb{F}_q$*:*

1. $C^\perp$ *is an* $[n, n-k]$*-linear code over* $\mathbb{F}_q$*.*
2. $(C^\perp)^\perp = C$*.*

**Definition .21 (Parity-Check Matrix)** *A* parity-check matrix $H$ *for an* $[n, k]$*-code* $C$ *is any generator matrix for* $C^\perp$ *[14].*

**Theorem .13 (Standard Form Parity-Check Matrix)** *If* $G = \left[I_k \mid A_{k\times(n-k)}\right]_{k\times n}$ *is a generator matrix for* $C$*, then:*

$$H = \left[-A^T_{(n-k)\times k} \mid I_{n-k}\right]_{(n-k)\times n},$$

*is a parity-check matrix for* $C$ *in standard form [14].*

### 3.1.2 Encoding and Decoding with Linear Codes

*Coding theory offers linear codes that are used for encoding and decoding messages. These codes can be viewed as invertible functions over vector spaces.*

### 3.1.3 Encoding Process

*Let* $\mathcal{C}$ *be an* $[n, k, d]$*–linear code over the finite field* $\mathbb{F}_q$*. Each codeword represents one piece of information, so* $\mathcal{C}$ *can represent* $q^k$ *distinct pieces of information. Suppose* $\mathbf{m} = (m_1, m_2, \ldots, m_k)$ *is a message in* $\mathbb{F}_q^k$*. Encoding is a one-to-one function:*

$$Encode : \mathbb{F}_q^k \longrightarrow \mathbb{F}_q^n,$$

*i.e., we append $n-k$ redundancy bits to the message $\mathbf{m}$ as:*

$$(m_1, m_2, \ldots, m_k) \mapsto (\hat{m}_1, \hat{m}_2, \ldots, \hat{m}_k, \hat{m}_{k+1}, \ldots, \hat{m}_n).$$

*Let $\mathcal{C}$ be an $[n,k]$–code over $\mathbb{F}_q$ with generator matrix $G$. Then $\mathcal{C}$ contains $q^k$ codewords and can represent $q^k$ messages. Encoding can be done via multiplication:*

$$\mathbf{m} \mapsto \mathbf{m} \cdot G.$$

*The resulting vector $\mathbf{m} \cdot G$ is a codeword of $\mathcal{C}$, being a linear combination of the generator matrix rows.*

**Example 12** ***Generator matrix of a binary Hamming code:***

- *The generator matrix $G$ of a binary $[7,4]$-Hamming code is given by*

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

- *The parameters of this code are $n = 7$, $k = 4$ and $d = 3$.*

- *The information word $u = (0,0,1,1)$ is encoded into the code word*

$$b = (0,0,1,1) \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} = (0,0,1,1,0,0,1).$$

**Definition .22** *Given a linear code $C \subseteq \mathbb{F}_q^n$ and a vector $\mathbf{u} \in \mathbb{F}_q^n$, the **coset** of $C$ shifted by $\mathbf{u}$ is:*

$$C + \mathbf{u} = \{\mathbf{v} + \mathbf{u} \mid \mathbf{v} \in C\}.$$

*All vectors in $C + \mathbf{u}$ are translations of codewords by $\mathbf{u}$ [14].*

**Theorem .14** *For an $[n, k, d]$-linear code $C$ over $\mathbb{F}_q$:*

*(i) Cosets cover the entire space: $\bigcup_{\mathbf{u}\in\mathbb{F}_q^n}(C+\mathbf{u}) = \mathbb{F}_q^n$.*

*(ii) Every coset is as large as $C$: $|C+\mathbf{u}| = q^k$.*

*(iii) If $\mathbf{u}$ is in $C+\mathbf{v}$ then, $C+\mathbf{u} = C+\mathbf{v}$.*

*(iv) Distinct cosets are non-overlapping.*

*(v) There are exactly $q^{n-k}$ unique cosets.*

*(vi) $\mathbf{u}$ and $\mathbf{v}$ share a coset iff $\mathbf{u}-\mathbf{v} \in C$.*

**Definition .23** *The **coset leader** is the vector with the smallest Hamming weight in its coset [10].*

## 3.2 Nearest Neighbor Decoding

*When a codeword $\mathbf{v}$ is sent and $\mathbf{w}$ is received, the error $\mathbf{e} = \mathbf{w}-\mathbf{v}$ lies in $\mathbf{w}+C$. To correct errors:*

1. *Locate $\mathbf{w}$ in the standard array.*
2. *Identify the coset leader $\mathbf{e}$ (smallest error).*
3. *Decode to $\mathbf{v} = \mathbf{w}-\mathbf{e}$.*

*Decoding maps $\mathbf{a}_i + \mathbf{c}_w$ to $\mathbf{c}_w$, with $\mathbf{a}_i$ as the error.*

**Example 13** *Let $C = \langle 1011, 0101\rangle$ be a linear code over $\mathbb{F}_2^4$.*

***Standard Array Construction:***

1. *Row 1 lists all codewords $C = \{\mathbf{c}_1, \ldots, \mathbf{c}_{q^k}\}$.*
2. *Add new rows by selecting the smallest unused vector $\mathbf{a}_i$ and computing $\mathbf{a}_i + C$.*
3. *Repeat until $\mathbb{F}_q^n$ is exhausted.*

*The standard array for $C$ is:*

$$C = \begin{Bmatrix} 0000 & 1011 & 0101 & 1110 \\ 1000 & 0011 & 1101 & 0110 \\ 0100 & 1111 & 0001 & 1010 \\ 0010 & 1001 & 0111 & 1100 \end{Bmatrix}.$$

- ***First row**: The codewords of $C$ (original code).*
- ***Subsequent rows**: Cosets formed by adding minimal-weight vectors:*
  - $1000 + C$ *(second row).*
  - $0100 + C$ *(third row).*
  - $0010 + C$ *(fourth row).*
- ***Coset leaders**: Leftmost vectors in each row (1000, 0100, 0010).*

## 3.3 Syndrome Decoding

*Standard array decoding works efficiently for codes with small length $n$, but becomes computationally intensive for larger $n$. Syndrome decoding optimizes this process by using coset identification via syndromes.*

### 3.3.1 Syndrome Fundamentals

**Definition .24 (Syndrome)** *For an $[n, k, d]$-linear code $C$ with parity-check matrix $H$, the* syndrome *of $\mathbf{w} \in \mathbb{F}_q^n$ is:*

$$S_H(\mathbf{w}) = \mathbf{w}H^T \in \mathbb{F}_q^{n-k}.$$

*(When $H$ is clear from context, we write simply $S(\mathbf{w})$.) [10].*

**Remark 7 (Syndrome Properties)** *For any $\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^n$:*

1. Linearity*: $S(\mathbf{u} + \mathbf{v}) = S(\mathbf{u}) + S(\mathbf{v})$.*
2. Codeword Test*: $S(\mathbf{y}) = \mathbf{0} \iff \mathbf{y} \in C$.*
3. Coset Identification*: $S(\mathbf{u}) = S(\mathbf{v}) \iff \mathbf{u} \equiv \mathbf{v} \pmod{C}$.*

### 3.3.2 Syndrome Look-Up Table Construction

*A* syndrome look-up table *pairs each coset leader with its syndrome:*

1. ***Coset Enumeration**:*
   - *List all distinct cosets of $C$ in $\mathbb{F}_q^n$ .*
   - *For each coset, select a minimum-weight vector $\mathbf{u}$ as coset leader.*

2. ***Syndrome Calculation**:*

   - *Compute* $S(\mathbf{u}) = \mathbf{u}H^T$ *for each coset leader.*
   - *Store pairs* (*Syndrome* $S(\mathbf{u})$, *Coset leader* $\mathbf{u}$).

   *This table enables efficient decoding by syndrome lookup rather than full coset searches.*

## 3.4 Syndrome Decoding Procedure

*Syndrome decoding operates on the same principles as standard array decoding but optimizes the process through syndrome matching:*

1. ***Syndrome Calculation**: For received vector* $\mathbf{w}$, *compute* $S(\mathbf{w}) = \mathbf{w}H^T$.
2. ***Coset Leader Lookup**: Find the coset leader* $\mathbf{u}$ *in the syndrome table where* $S(\mathbf{u}) = S(\mathbf{w})$.
3. ***Decoding**: Recover the codeword* $\mathbf{v} = \mathbf{w} - \mathbf{u}$.

*This method eliminates the need for exhaustive search through the standard array, significantly improving decoding efficiency.*

**Example 14** *Let* $C$ *be* $[6, 3, 3]$*-linear code with generator matrix*

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

*and parity check matrix*

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

*The syndromes and coset leaders are:*

| *Syndrome* | *Coset Leaders* |
|---|---|
| 000 | $(0,0,0,0,0,0)$ |
| 011 | $e_1$ |
| 101 | $e_2$ |
| 110 | $e_3$ |
| 100 | $e_4$ |
| 010 | $e_5$ |
| 001 | $e_6$ |
| 111 | $e_1+e_4, e_2+e_5, e_3+e_6$ |

*such that* $e_1=(1,0,0,0,0,0),\cdots,e_6=(0,0,0,0,0,1)$.

*If we receive the vector* $v=(111101)$. *then* $v=c+e$ *(*$c$ *is the transmitted code-word and* $e$ *is the error vector).*

*We have that :*

$$eH^T=(v-c)H^T=vH^T-cH^T=vH^T-0=vH^T=(101).$$

*From the table* $e$ *(and* $v$*) are in the coset with leader* $e_2$.

*Under nearest-neighbor decoding, we want wt(*$e$*) to be as small as possible, so we assume* $e=e_2$. *Then* $c=v-e_2=(111101)-(010000)=(101101)$.

*The original message is the information symbols in* $c$ *(the first three positions), or* 101. *If we receive the vector* $v=c+e=(100100)$.

*We have that* $vH^T=(111)$, *and* $e$ *is in the coset of* $e_1+e_4$.

*But there are three equally likely alternatives for the error vector. We have detected two errors, but we cannot correct them. We can always correct one error in a block, and occasionally we can detect two errors (we can detect but not correct the errors).*

## 3.5 Goppa Codes

### 3.5.1 Historical Context

*Discovered in 1970 by Valery Goppa, these codes established a fundamental connection between algebraic geometry and coding theory. They form an important subclass of alternant codes and feature efficient decoding algorithms.*

**Definition .25 (Goppa Code)** *[14] Let:*

- $g(z) = \sum_{i=0}^{t} g_i z^i \in \mathbb{F}_{q^m}[z]$ *be a* Goppa polynomial . *Where $t$ represents the degree of the polynomial $g(z)$ and determines the number of errors the code can correct.*
- $L = \{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}_{q^m}$ *with $g(\alpha_i) \neq 0$ for $i = 1, ..., n$.*

*The* Goppa code $\Gamma(L, g(z))$ *is:*

$$\left\{ \mathbf{c} \in \mathbb{F}_q^n \;\middle|\; \sum_{i=1}^{n} \frac{c_i}{z - \alpha_i} \equiv 0 \mod g(z) \right\}.$$

### 3.5.2 Key Properties

**Theorem .15 (Inverse Computation)** *[14] In $\mathbb{F}_{q^m}[z]/\langle g(z) \rangle$:*

$$(z - \alpha_i)^{-1} = -\left( \frac{g(z) - g(\alpha_i)}{z - \alpha_i} \right) g(\alpha_i)^{-1}.$$

**Corollary 2 (A vector $c \in \Gamma(L, g)$ if and only if)**

$$\sum_i c_i \left( \frac{g(x) - g(\alpha_i)}{x - \alpha_i} \right) g(\alpha_i)^{-1} = 0 \text{ as a polynomial in } \mathbb{F}_q[x]$$

*Hence, we derive the Parity check matrix over $\mathbb{F}_q$ for the Goppa codes as.*

**Corollary 3** *For a Goppa code $\Gamma(L, g(z))$, the Parity check matrix over $\mathbb{F}_q$ is $H$*

$$H = \begin{pmatrix} g_t g(\alpha_1)^{-1} & g_t g(\alpha_2)^{-1} & \cdots & g_t g(\alpha_n)^{-1} \\ (g_t \alpha_1 + g_{t-1}) g(\alpha_1)^{-1} & (g_t \alpha_2 + g_{t-1}) g(\alpha_2)^{-1} & \cdots & (g_t \alpha_n + g_{t-1}) g(\alpha_n)^{-1} \\ \vdots & \vdots & \ddots & \vdots \\ (g_t \alpha_1^{t-1} + \cdots + g_1) g(\alpha_1)^{-1} & (g_t \alpha_2^{t-1} + \cdots + g_1) g(\alpha_2)^{-1} & \cdots & (g_t \alpha_n^{t-1} + \cdots + g_1) g(\alpha_n)^{-1} \end{pmatrix}.$$

*This matrix is further broken down into product of three matrices as:*

$$H = \underbrace{\begin{pmatrix} g_t & 0 & \cdots & 0 \\ g_{t-1} & g_t & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_1 & g_2 & \cdots & g_t \end{pmatrix}}_{C} \cdot \underbrace{\begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{t-1} & \alpha_2^{t-1} & \cdots & \alpha_n^{t-1} \end{pmatrix}}_{X} \cdot \underbrace{\begin{pmatrix} g(\alpha_1)^{-1} & 0 & \cdots & 0 \\ 0 & g(\alpha_2)^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g(\alpha_n)^{-1} \end{pmatrix}}_{Y}.$$

*Now, as we have $c \in \Gamma(L, g)$ if and only if $cH^T = 0$, which implies $c(CXY)^T = 0$, equivalently $cY^T X^T C^T = 0$, and this gives $cY^T X^T = 0$, or $c(XY)^T = 0$ (as matrix $C$ is invertible).*

**Remark 8** *The matrix $XY$ can be viewed as Parity check matrix for $\Gamma(L, g)$ over $\mathbb{F}_q$. The matrix*

$$XY = \begin{pmatrix} g(\alpha_1)^{-1} & g(\alpha_2)^{-1} & \cdots & g(\alpha_n)^{-1} \\ \alpha_1 g(\alpha_1)^{-1} & \alpha_2 g(\alpha_2)^{-1} & \cdots & \alpha_n g(\alpha_n)^{-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{t-1} g(\alpha_1)^{-1} & \alpha_2^{t-1} g(\alpha_2)^{-1} & \cdots & \alpha_n^{t-1} g(\alpha_n)^{-1} \end{pmatrix}_{t \times n}.$$

**Remark 9** *Viewing elements of $\mathbb{F}_{q^m}$ as vectors of length $m$ over $\mathbb{F}_q$ by vector space isomorphism, we have a Parity check matrix for $\Gamma(L, g)$ over $\mathbb{F}_q$ to be an $m \cdot t \times n$ matrix, with at least $t$ columns linearly independent over $\mathbb{F}_q$. Hence, the Hamming distance of Goppa code, $d(\Gamma(L, g)) \geq t + 1$. Since, for the matrix $XY$ over $\mathbb{F}_q$, maximum of $mt$ rows are linearly independent, hence, Rank$(XY) \leq mt$, which gives Nullity$(XY) \geq n - mt$. Therefore, dimension of Goppa code, $\dim_{\mathbb{F}_q} \Gamma(L, g) \geq n - mt$.*

**Example 15** *(A Goppa Code) Let $\mathbb{F}_{2^4}$ be the field isomorphic to $\frac{\mathbb{F}_2[x]}{\langle x^4+x+1 \rangle}$. Let $\alpha$ be a root of $x^4 + x + 1$, then, since multiplicative order of $\alpha$ is 15, it can be used to generate all elements of $\mathbb{F}_{2^4}$, equivalently, $\mathbb{F}_{2^4} = \langle \alpha \rangle$. Hence we can represent elements of $\mathbb{F}_{2^4}$ as:*

$$
\begin{array}{lcll}
0 & = & 0 & = (0,0,0,0)^T, \\
1 & = & 1 & = (1,0,0,0)^T, \\
\alpha & = & \alpha & = (0,1,0,0)^T, \\
\alpha^2 & = & \alpha^2 & = (0,0,1,0)^T, \\
\alpha^3 & = & \alpha^3 & = (0,0,0,1)^T, \\
\alpha^4 & = & 1+\alpha & = (1,1,0,0)^T, \\
\alpha^5 & = & \alpha+\alpha^2 & = (0,1,1,0)^T, \\
\alpha^6 & = & \alpha^2+\alpha^3 & = (0,0,1,1)^T, \\
\alpha^7 & = & 1+\alpha+\alpha^3 & = (1,1,0,1)^T, \\
\alpha^8 & = & 1+\alpha^2 & = (1,0,1,0)^T, \\
\alpha^9 & = & \alpha+\alpha^3 & = (0,1,0,1)^T, \\
\alpha^{10} & = & 1+\alpha+\alpha^2 & = (1,1,1,0)^T, \\
\alpha^{11} & = & \alpha+\alpha^2+\alpha^3 & = (0,1,1,1)^T, \\
\alpha^{12} & = & 1+\alpha+\alpha^2+\alpha^3 & = (1,1,1,1)^T, \\
\alpha^{13} & = & 1+\alpha^2+\alpha^3 & = (1,0,1,1)^T, \\
\alpha^{14} & = & 1+\alpha^3 & = (1,0,0,1)^T.
\end{array}
$$

*Consider the Goppa Code* $\Gamma(L, g(z))$ *defined by:*

$$g(z) = (z+\alpha)(z+\alpha^{14}) = z^2 + \alpha^7 z + 1.$$

$$L = \left\{\alpha^i \mid 2 \leq i \leq 13\right\} \ ; |L| = 12, \quad \alpha^i \neq \alpha, \alpha^{14}.$$

*Now, in order to find the Parity check matrix of* $\Gamma(L, g(z))$*, we need to compute* $g(\alpha_i)^{-1}$ *such that* $\alpha_1 = \alpha^2, \alpha_2 = \alpha^3, \cdots, \alpha_{12} = \alpha^{13}$ .
*The factors* $g(\alpha_i)^{-1}$ *;* $1 \leq \alpha_i \leq 12$ *are computed as follow:*

$$g(\alpha^2)^{-1} = (\alpha^4+\alpha^9+1)^{-1} = ((1,1,0,0)^T+(0,1,0,1)^T+(1,0,0,0)^T)^{-1} = (0,0,0,1)^T = (\alpha^3)^{-1} = \alpha^{12}$$

$$H = \begin{pmatrix} \alpha^9 & \alpha^{10} & \alpha^9 & \alpha^{14} & \alpha^6 & 0 & \alpha^{10} & \alpha^8 & \alpha^2 & \alpha^7 & \alpha^{14} & \alpha^6 \\ \alpha^{12} & \alpha^6 & \alpha^6 & \alpha & \alpha^{11} & 1 & \alpha^{14} & \alpha^8 & \alpha^{11} & \alpha^{14} & \alpha^{12} & \alpha \end{pmatrix}.$$

*The minimum distance of* $C$ *is* 3.

*The entries can be observed as binary vectors of length 4 using the table described above. This is given by:*

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

*Then, the Null-space of $H$ produces generator matrix of $(L; g(z))$. Hence the generator matrix $G$ becomes:*

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

### 3.5.3 Encoding with Goppa Codes

*Let $\Gamma(L, g(z))$ be a Goppa code, where $g(z)$ is a primitive polynomial with $\deg(g(z)) = t$ and $|L| = n$. Let $\dim_{F_q}(\Gamma(L, g(z))) = k$, and $G$ be a $k \times n$ generator matrix for the respective Goppa code. The encoding of a $k$-length message vector $\mathbf{m}$ over $F_q$ is given by:*

$$\mathbf{c} = \mathbf{m}G.$$

### 3.5.4 Correction of Errors / Syndrome Decoding of Goppa Codes

*Let the vector $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ be received with $r$ errors, where $2r + 1 \leq d$ (for maximum error correction capability). Let $L = \{a_1, a_2, \ldots, a_n\}$. The received vector can be expressed as:*

$$\mathbf{y} = (y_1, y_2, \ldots, y_n) = \mathbf{c} + \mathbf{e} = (c_1, c_2, \ldots, c_n) + (e_1, e_2, \ldots, e_n),$$

*where $\mathbf{c}$ is the codeword and $\mathbf{e}$ is the error vector with $e_i \neq 0$ at exactly $r$ positions. The goal is to:*

- *Locate the error positions $B = \{i : 1 \leq i \leq n \text{ and } e_i \neq 0\}$.*
- *Find the corresponding error values $e_i$ for $i \in B$.*

*To achieve this, we define two polynomials.*

**Definition .26 (Error Locator and Evaluator Polynomials)** *[14]*

- *The* ***error locator polynomial***

  $\sigma(z)$ *is defined as:*

$$\sigma(z) := \prod_{j \in B} (z - \alpha_j) \quad \textit{(a polynomial of degree } r\textit{)}.$$

- *The* ***error evaluator polynomial*** $w(z)$ *is defined as:*

$$w(z) := \sum_{i \in B} e_i \prod_{\substack{j \in B \\ j \neq i}} (z - \alpha_j) \quad \textit{(a polynomial of degree } r\textit{)}.$$

**Definition .27 (Syndrome of Received Vector)** *[14] The* ***syndrome*** *of the received vector* $\mathbf{y}$ *is defined as:*

$$S(y) := \sum_{i=1}^{n} \frac{y_i}{z - \alpha_i} \equiv \sum_{i \in B} \frac{e_i}{z - \alpha_i} \pmod{g(z)}.$$

**Proposition .1** *Let* $\mathbf{e}$ *be the error vector having weight* $r$ *where* $r \leq \lfloor \frac{t}{2} \rfloor$*. Let* $\sigma(z)$*,* $w(z)$ *and* $S(y)$ *be as described above. Then the following properties hold:*

*(i)* $\deg(\sigma(z)) = r$.

*(ii)* $\deg(w(z)) \leq r - 1$.

*(iii)* $\gcd(\sigma(z), w(z)) = 1$.

*(iv)* $e_k = \frac{w(\alpha_k)}{\sigma'(\alpha_k)}$*, where* $k \in B$ *and* $\sigma'$ *represents the derivative of* $\sigma$.

*(v)* $\sigma(z)S(y) \equiv w(z) \pmod{g(z)}$.

### 3.5.5 Error-correction Algorithm

*Algorithm for correcting* $r \leq \lfloor \frac{t}{2} \rfloor$ *errors in a Goppa code:*

*Step (i): Compute the syndrome*

$$S(y) = \sum_{i=1}^{n} \frac{y_i}{z - \alpha_i}.$$

*Step (ii): Solve the key equation*

$$\sigma(z)S(y) \equiv w(z) \pmod{g(z)},$$

*by writing*

$$\sigma(z) = \sigma_0 + \sigma_1 z + \cdots + \sigma_{r-1} z^{r-1} + z^r,$$
$$w(z) = w_0 + w_1 z + \cdots + w_{r-1} z^{r-1}$$

*and solve for $t$ equations and $2r$ unknowns.*

*If the code is binary, take $w(z) = \sigma'(z)$.*

*Step (iii):* *Determine the set of error locations $B = \{i : 1 \leq i \leq n \text{ and } \sigma(\alpha_i) = 0\}$.*

*Step (iv):* *Compute the error values $e_i = \frac{w(\alpha_i)}{\sigma'(\alpha_i)}$ for all $i \in B$.*

*Step (v):* *The error vector $\mathbf{e} = (e_1, e_2, \ldots, e_n)$ is defined by $e_i$ for $i \in B$ and zeros elsewhere.*

*Step (vi):* *The codeword sent is calculated as $\mathbf{c} = \mathbf{y} - \mathbf{e}$.*

### 3.5.6 Patterson's Algorithm for Error Correction

*The Patterson algorithm decodes only binary Goppa codes. It computes the syndrome $S(y)$ of a received vector and then solves the key equation $\sigma(z)S(y) \equiv w(z) \pmod{g(z)}$ with $w(z) = \sigma'(z)$ by heavily exploiting the requirement that the code is binary. The error locator polynomial can be split in even and odd powers of $z$ such that $\sigma(z) = a^2(z) + zb^2(z)$, as field has characteristic 2.*

*The Patterson algorithm can be described as below:*

***Input:*** *The received vector $\mathbf{y}$ and the Goppa code $\Gamma(L; g)$.*

*Step (i):* *Compute syndrome $S(y)$ an element of $\mathbb{F}_{q^m}[z]/\langle g(z)\rangle$;*

*Step (ii):* *Compute $T(z) = S(y)^{-1} \mod g(z)$.*

*Step (iii):* *Compute $P(z) = \sqrt{T(z) + z} \mod g(z)$.*

*Step (iv):* *Compute $u(z)$ and $v(z)$ with $u(z) = v(z)S(y) \mod g(z)$.*

*Step (v):* *Compute the locator polynomial $\sigma(z) = u(z)^2 + zv(z)^2$.*

*Step (vi):* *Find the roots of $\sigma(z)$.*

*Step (vii):* *Find error positions, i.e., error vector $\mathbf{e}$.*

***Output:*** *The error vector $\mathbf{e}$.*

## 3.6 Decoding Process

*After correcting possible errors in a codeword, one can find the message sent by recalling that*

$$(m_1, m_2, \ldots, m_k) \cdot G = (c_1, c_2, \ldots, c_n),$$

*equivalently,*

$$G^T \cdot \begin{pmatrix} m_1 \\ \vdots \\ m_k \end{pmatrix} = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}.$$

*To find the message vector $(m_1, m_2, \ldots, m_k)$, one reduces the above system to*

$$\left[\begin{array}{c|c} & c_1 \\ G^T & \vdots \\ & c_n \end{array}\right] \sim \left[\begin{array}{c|c} & m_1 \\ I_k & \vdots \\ & m_k \\ \hline 0 & P \end{array}\right].$$

**Example 16** *Let $\mathbb{F}_{3^2}$ be the field corresponding to the primitive polynomial $x^2 - x - 1$ over the base field $\mathbb{F}_3$ and let $\alpha$ be one of its roots. Then we have:*

$$\begin{aligned} 0 = \mathbf{0} &= (0,0)^T, & \alpha^3 &= 1 - \alpha = (1,-1)^T, \\ 1 = \mathbf{1} &= (1,0)^T, & \alpha^4 &= -1 = (-1,0)^T, \\ \alpha &= \alpha = (0,1)^T, & \alpha^5 &= -\alpha = (0,-1)^T, \\ \alpha^2 &= 1 + \alpha = (1,1)^T, & \alpha^6 &= -1 - \alpha = (-1,-1)^T, \\ & & \alpha^7 &= -1 + \alpha = (-1,1)^T. \end{aligned}$$

*Consider the Goppa code $\Gamma(L, g(z))$ defined by:*

$$g(z) = z(z - \alpha^7) = z^2 + \alpha^3 z, \quad L = \{\alpha^i : 0 \leq i \leq 6\}.$$

*The parity check matrix over $\mathbb{F}_3(\alpha)$ is:*

$$H = \begin{pmatrix} \alpha^4 & \alpha^3 & \alpha^2 & \alpha & 1 & \alpha^7 & \alpha^6 \\ \alpha^6 & \alpha^3 & \alpha^6 & \alpha^2 & \alpha^3 & \alpha^5 & \alpha^5 \end{pmatrix}.$$

*Equivalently, the parity check matrix over $\mathbb{F}_3$ is:*

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

*This gives the generator matrix as:*

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

*The parameters of this code are* $[7, 3, \geq 3]$.

*Now, let the message* $\mathbf{m} = (0, 0, 0)$ *be sent. Firstly, encoding of this vector will be* $\mathbf{c} = \mathbf{m}G = (0, 0, 0, 0, 0, 0, 0)$. *Suppose that the vector* $\mathbf{y} = (0, 0, 0, 0, 0, 0, -1)$ *is received having one error. Our aim is to find the error vector.*

*(i)* ***Syndrome****:*

$$\begin{aligned} S(y) &= \sum_{i=1}^{6} \frac{y_i}{z - \alpha^i} = \frac{-1}{z - \alpha^6} \\ &\equiv \alpha^2 + \alpha z \mod g(z). \end{aligned}$$

*(ii) Substituting* $\sigma(z) = \sigma_0 + z$ *and then computing* $\sigma(z)S(y) \mod z^2 + \alpha^3 z$ *gives:*

$$\begin{aligned} \sigma(z)S(y) &= (\sigma_0 + z)(\alpha^2 + \alpha z) \\ &= \sigma_0\alpha^2 + (\alpha^2 + \alpha\sigma_0)z + \alpha z^2 \\ &\equiv \alpha^2\sigma_0 + (\alpha^2 + \alpha\sigma_0\alpha^4)z \\ &= \alpha^2\sigma_0 + (\alpha^7 + \alpha\sigma_0)z. \end{aligned}$$

*Thus, for* $w(z) = w_0$, *we get the system of equations by comparing coefficients of* $\sigma(z)S(y) \equiv w(z) \mod g(z)$ *as:*

$$\begin{cases} w_0 = \alpha^2\sigma_0, \\ 0 = \alpha^7 + \alpha\sigma_0. \end{cases}$$

*The solution to this system is* $\sigma_0 = \alpha^2$, $w_0 = \alpha^4$ *and hence* $\sigma(z) = z + \alpha^2$ *and* $w(z) = \alpha^4$.

*(iii) The root of* $\sigma(z)$ *is* $\alpha^6 = \alpha^7$, *thus the set of error locations is:*

$$B = \{i \mid \sigma(\alpha^i) = 0\} = \{7\}.$$

*(iv) The error value:*

$$e_7 = \frac{\alpha^4}{1} = \alpha^4 = -1.$$

*(v) The codeword sent must have been:*

$$\mathbf{c} = \mathbf{y} - \mathbf{e} = (0, 0, 0, 0, 0, 0, 0).$$

*(vi) The original message can be found by solving the augmented system:*

$$\left[\begin{array}{c} G^T \\ \hline \mathbf{c}^T \end{array}\right],$$

*which results in* $(0, 0, 0)$.

# Chapter 4

# McEliece Public Key Encryption Cryptosystem

This chapter introduces the McEliece cryptosystem, a public key encryption scheme based on error-correcting codes. It begins by explaining the probabilistic key generation method, followed by the encryption and decryption processes. The chapter also discusses possible attacks on the system and evaluates its strengths and weaknesses, particularly in the context of post-quantum security.For more information, see [3] [4] [8] [9] [11] [12] [13] [16].

The McEliece Public Key Cryptosystem pioneered the use of randomization in encryption and relies on the complexity of decoding linear codes, an NP-hard problem (Non-Polynomial Deterministic Time Hard).

In the original design, the private key is constructed using binary Goppa codes, though other code families can be employed in modified versions. However, such alternatives may not offer the same security guarantees as Goppa codes.

The original McEliece scheme:

- Forms the foundation of the NIST submission "Classic McEliece".
- Introduced the paradigm of hiding structures of efficiently decodable linear codes.
- Inspired numerous modern code-based cryptosystems.

1. **Code Variants**:
    - Attempts to replace Goppa codes failed due to structural vulnerabilities.
    - Original Goppa-based version remains most secure.

- Exploration of rank/Lee metric codes emerged.

2. **Cryptanalysis**:

   - Best attacks use *Information Set Decoding (ISD)* .
   - Complexity remains exponential even with quantum computers .
   - Security maintained via parameter updates.

## 4.1 A Probabilistic Key Generation Algorithm

The probabilistic key generation mechanism at the heart of the McEliece cryptosystem relies on the algebraic structure of Goppa codes, combined with random linear transformations, to produce a secure public-private key pair. The process involves selecting code parameters, applying scrambling and permutation matrices, and constructing a disguised generator matrix. These steps ensure both the correctness of encryption and decryption and the security of the system against structural attacks.

### 4.1.1 Key Generation Algorithm

The key generation mechanism employed in the McEliece cryptosystem is built upon structured randomness and leverages the algebraic properties of binary Goppa codes. It serves as the foundation for constructing secure and efficient public-private key pairs.

- A **public key**: A structured but disguised generator matrix.
- A **private key**: A binary irreducible Goppa code with trapdoor information.

### 4.1.2 Core Components

The fundamental elements constituting both the public and private keys are presented in this part. Each component is described in terms of its structure and cryptographic role.

- **Public Key** $(G_{pub}, t)$:
  - $G_{pub}$: A transformed generator matrix $(k \times n)$.
  - $t$: Maximum number of correctable errors.
- **Private Key** $(S, G, P)$:

- $S$: Random non-singular scrambling matrix ($k \times k$).
- $G$: Original generator matrix of the Goppa code.
- $P$: Random permutation matrix ($n \times n$).

### 4.1.3 Key Generation Steps

This part outlines the sequential steps involved in constructing the key pair, from the initial choice of parameters to the derivation of the public key through algebraic transformations.

1. **Select Parameters**:
   - Choose a binary $(n, k)$ linear code $C$ correcting $t$ errors (with $t \leq n$).
   - Ensure $C$ has an efficient decoder and minimum distance $d \geq 2t + 1$.
2. **Construct Matrices**:
   - $G$: Generator matrix ($k \times n$) of $C$.
   - $S$: Random invertible matrix ($k \times k$).
   - $P$: Random permutation matrix ($n \times n$).
3. **Compute Public Key**:
$$G_{pub} = S \cdot G \cdot P.$$

### 4.1.4 Encryption Process

The encryption procedure is detailed, illustrating how a plaintext message is encoded using the public key and masked with a controlled random error vector .

- **Plaintext** ($m$): A binary vector of length $k$.
- **Ciphertext** ($c$): Computed as:
$$c = m \cdot G_{pub} + e,$$
where $e$ is a random error vector with $\leq t$ bits set to 1.

### 4.1.5 Decryption Process

In this part we describe the decryption phase, in which the legitimate receiver applies inverse transformations and decoding techniques to recover the original message.

1. **Invert Permutation**: Compute $c' = c \cdot P^{-1}$.

2. **Decode**: Use the Goppa code decoder to remove errors and recover $m \cdot S$.

3. **Descramble**: Multiply by $S^{-1}$ to obtain the original message $m$.

### 4.1.6 Security Properties

The main security characteristics of the McEliece cryptosystem are highlighted, particularly its reliance on the computational hardness of decoding random linear codes and the effectiveness of Goppa code-based decoding.

- **Security**: Relies on the hardness of decoding random linear codes (NP-hard).
- **Efficiency**: Goppa codes enable fast decoding with the private key.
- **Randomization**: The scrambling ($S$) and permutation ($P$) hide the structure of $G$.

### 4.1.7 Probabilistic Encryption Algorithm

This final part presents the complete probabilistic encryption algorithm, emphasizing the mathematical structure and randomness that ensure confidentiality and robustness against cryptanalysis.

1. **Message Encoding**:
   - Encode the message $M$ as a binary string of length $k$.
2. **Error Vector Generation**:
   - Generate a random vector $z \in \mathbb{F}_2^n$ of length $n$ and Hamming weight $t$.
   - (Each $z$ has exactly $t$ non-zero elements).
3. **Ciphertext Computation**:

$$C_t = M \cdot G_{pub} + z,$$

   where:
   - $C_t$ is the resulting ciphertext.
   - $G_{pub}$ is the recipient's public key matrix.
   - The addition is performed modulo 2.

## 4.2 Properties of the Encryption

- **Probabilistic Nature**: Each encryption uses a fresh random $z$, making identical plaintexts encrypt to different ciphertexts.
- **Security**: The random error vector $z$ ensures security against chosen plaintext attacks.
- **Error Correction**: The recipient can correct up to $t$ errors using their private key.

## 4.3 Decryption Algorithm

Upon receiving ciphertext $C_t$, the receiver performs the following steps to recover the original message:

1. **Invert Permutation**:
   - Compute the inverse of $P$: $P^{-1}$.
   - Calculate:
   $$Code_t = C_t \cdot P^{-1} = (M \cdot G_{pub} + z) \cdot P^{-1}.$$
2. **Expand the Calculation**:
   $$Code_t = M \cdot G_{pub} \cdot P^{-1} + z \cdot P^{-1} = M \cdot S \cdot G \cdot P \cdot P^{-1} + z \cdot P^{-1} = M \cdot S \cdot G + z',$$
   where $z' = z \cdot P^{-1}$ has the same weight $t$ as $z$.
3. **Decode**:
   - Apply the decoding algorithm for $G$ (which can correct up to $t$ errors).
   - The codeword $M \cdot S \cdot G$ is at Hamming distance $t$ from $Code_t$.
   - Recover $M \cdot S$.
4. **Descramble**:
   $$M = M \cdot S \cdot S^{-1},$$
   obtaining the original plaintext message

### 4.3.1 Security Parameters

The McEliece cryptosystem uses parameters $[n, k, d]$ where:

- $n$: Code length.
- $k$: Dimension.
- $d$: Hamming distance of weight $t$.

The dimension is calculated as:

$$k = n - m \cdot t.$$

Robert J. McEliece originally suggested using binary ($p = 2$) Goppa codes with:

- $m = 10$.
- $n = 1024$.
- $t = 50$.

**Example 17** *We will use the generator matrix of a Goppa code over the subset* $L = \{\alpha^i \text{ such that } 2 \leq i \leq 3\}$ *with* $g(x) = x^2 + x + \alpha^3$ *(over the field* $\mathbb{F}_2[x]/x^4 + x + 1$*) given by:*

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

*We get the matrix* $G$ *using the same method in ( Definition .18 and Example 15).*

*We choose a random matrices* $S$ *and* $P$ *as the following:*

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

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

*Using the two matrices, we compute the public encryption matrix as follows:*

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

*Using this matrix we can encrypt a mesage. We suppose that we want to send the message* $m = (1, 0, 1, 0)$*, then we compute the matrix:*

$$mG^{'} = (1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0),$$

*we add a random error vector* $e = (1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ *so that:*

$$y = mG^{'} + e = (0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0),$$

*this results the ciphertext, to get the original message we first compute:*

$$yP^{-1} = mG^{'}P^{-1} + eP^{-1} = (mS)G + e^{'} = (1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0).$$

*With this permutation the errors have been moved to the first and sixth columns, to correct these we use the error-correcting algorithm to find:*

$$mSG = (1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0),$$

*by row reduction we find* $mS = (1, 1, 0, 1)$*, to find the plaintext we compute*

$$m = (1, 1, 0, 1) \cdot \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} = (1, 0, 1, 0).$$

## 4.4 Attacking

In cryptology, there are various types of attacks. A **brute force attack** is when the enemy tries every possible key and determines which key results in a meaningful message. For this attack, the longer the length of a key, the longer it will take to try every possible key. A **direct**, or **per-message**, attack is an attack that tries to decode a given message, but does not necessarily solve the entire cryptosystem. A **structural attack** occurs when Eve would try to recover the structure, or at least part of the structure, of the original message from the public key.

The most effective attack against the McEliece cryptosystem is an attack called **"information-set decoding"**. Many cryptologists have published variants of this attack from McEliece in his original message [K] to Stern [M] in his subsequent message. Most of the variants of the information-set decoding are based off of Stern's attack.

Stern's attack refers to Stern's method on finding codewords with a low Hamming weight. In order to do so, we must select an integer $w \geq 0$ and a $(n - k) \times n$ parity check matrix $K$ for an $[n, k]$ code over the binary field. Our goal is to find a solution to the equation $Kz = 0$ with the weight $|z| = w$. Then we complete the following steps:

1. Randomly select a $n \times k$ column from $K$. Select $n - k - 1$ additional columns out of the remaining columns of $K$ such that each column is chosen due to pivots in previous columns by using Gaussian elimination.

2. Randomly select a subset $Z$ from the $n - k$ columns consisting of $l$ elements.

3. Partition the leftover columns into two subsets $X$ and $Y$. Each column is added to either $X$ or $Y$ with the probability of $\frac{1}{2}$.

4. Search for codewords with exactly $p$ non-zero bits in $X$, exactly $p$ non-zero bits in $Y$, no non-zero bits in $Z$, and exactly $w - 2p$ non-zero bits in the remaining columns by completing the following steps:

(a) Apply elementary row operations to the matrix $K$ so that the randomly selected $n-k$ columns become the identity matrix.

(b) For every subset $A$ of $X$ with $p$ elements, add the columns of the matrix matrix 2 for each row $l$ in order to compute the $l$-bit vector $\pi(A)$. Do the same for every subset $B$ of $Y$ with $p$ elements.

(c) Find all pairs $A, B$ such that $\pi(A) = \pi(B)$. For every such pair, compute the sum of all members of $A \cup B$. Said sum is a $(n-k)$-bit vector. If the vector does not have a weight of $w - 2p$, then the attack has failed. If the vector does have a weight of $w - 2p$, then the solution can be found by adding the corresponding columns and those columns, together with $A$ and $B$, form a codeword with weight $w$. If there are no such codewords, the attack fails and Stern starts over with s new selection of columns.

Say we have a code $C$ over the binary field. Let $y$ have a distance $w$ from a codeword $c \in C$, then $y - c$ is a element with a weight w of the code $C + \{0, y\}$. Eve knows the McEliece public key $G', t$. thus, Eve can add y to the list of generators in order to form a generator matrix $C + \{0, y\}$. Note that the only codeword with weight $t$ is $y - c$, which is precisely the codeword we found with Stern's attack. Therefore, Eve can use this codeword to find $c$ and solve for the original message.

## 4.5 Strengths and Weaknesses

### 4.5.1 Strengths

1. **Speed**: 2-3 orders faster than RSA (original paper).
2. **Quantum Resistance**: Withstands Shor's algorithm.
3. **Continuous Improvement**: Ongoing optimizations.

### 4.5.2 Weaknesses and Criticisms

1. **Large Keys**:
   - Public key: $\mathcal{O}(n^2)$ bits.
   - Low-rate codes required ($R = k/n < 0.5$).
2. **CCA Vulnerability**:
   - Vulnerable to chosen-ciphertext attacks.

- Adversary can modify ciphertext bits.

### 4.5.3 McEliece Today

- **NIST Mitigation**: "Classic McEliece" returns fake plaintexts.
- **Real-World Use**: "iOS app" PQChat.
- **Parameters**:

| Security | $n$ | $k$ | Key size (KB) |
|---|---|---|---|
| 128-bit | 3488 | 2720 | 261 |
| 256-bit | 6624 | 5129 | 1362 |

Table 4.1: Modern McEliece parameters

# *Conclusion*

This thesis presents a comprehensive study of cryptography, with a focus on **code-based cryptography** and **Goppa codes**, which form the basis of the **McEliece cryptosystem**. It begins with an overview of *symmetric and asymmetric algorithms*, followed by the *algebraic foundations* such as finite fields. The core of the work examines the *construction and decoding* of Goppa codes, including *Patterson's algorithm* and their *resistance to attacks* like Stern's. The study emphasizes the dual role of Goppa codes in *error correction* and *post-quantum security*, confirming their importance in modern cryptography.

# Bibliography


[1] Algebra, Artin, M., *Prentice Hall*, Inc. , Englewood Cliffs, NJ, 1991.

[2] Ben-Sasson, Eli and Carmon, Dan and Kopparty, Swastik and Levit, David, *Elliptic curve fast Fourier transform ECFFT part I: low-degree extension in time O n log n over all finite fields*, Annual ACM-SIAM Symposium on Discrete Algorithms SODA, 700–737, 2023.

[3] Diffie, Whitfield and Hellman, Martin,E., *New directions in cryptography*, Democratizing cryptography: the work of Whitfield Diffie and Martin Hellman, 365–390, 2022.

[4] Faugere, Jean-Charles and Gauthier-Umana, Valérie and Otmani, Ayoub and Perret, Ludovic and Tillich, Jean-Pierre, *A distinguisher for high-rate McEliece cryptosystems*, IEEE Transactions on Information Theory, **59**, (10), 6830–6844, 2013.

[5] Hoffstein, Jeffrey and Pipher, Jill and Silverman, Joseph H, *An introduction to cryptograph*, An Introduction to Mathematical Cryptography, 1–59, 2014.

[6] Kaur, Ravinder and Sahu, Chinmay, *Cryptography in industry*, Next Generation Mechanisms for Data Encryption, 146, 2025.

[7] Kessler, Gary,C., *An overview of cryptography*, 2003.

[8] Kimura, Naoto and Takayasu, Atsushi and Takagi, Tsuyoshi, *Memory-efficient quantum information set decoding algorithm*, Australasian Conference on Information Security and Privacy, 452–468, 2023.

[9] McEliece, Robert,J., *A public-key cryptosystem based on algebraic*, Coding Thv, **4244**, (1978), 114–116, 1978.

[10] Patterson, Nicholas, *The algebraic decoding of Goppa codes*, IEEE Transactions on Information Theory, **21**, (2), 203–207, 1975.

[11] Prange, Eugene, *The use of information sets in decoding cyclic codes*, IRE Transactions on Information Theory, **8**, (5), 5–9, 1962.

[12] Rakotondrafara, Antsa, *A Survey of Code-Based Cryptography*, 2023.

[13] Rivest, Ronald,L., and Shamir, Adi and Adleman, Leonard, *A method for obtaining digital signatures and public-key cryptosystems*, Communications of the ACM, **26**, (1), 96–99, 1983.

[14] Singh, Harshdeep, *Code based cryptography: Classic mceliece*, arXiv preprint arXiv:1907.12754, 2019.

[15] Smart, Nigel,P., *Cryptography made simple*, 2016.

[16] Valentijn, Ashley, *Goppa codes and their use in the mceliece cryptosystems*, 2015.