Introduction to Algebraic Codes

T.T.Moh
Mathematics Department
Purdue University
W.Lafayette, Indiana

To
Ping

Preface

In the 19-th century, *Applied Mathematics* are *continous* and sometime *differenciable*. One of the important topics is the *differential equations*. Laplace once said that God must be an expert in solving differential equations. In the 20th century, the creation of computers made the large scale arithmetical operations sinpler, hence the *Applied Finite Mathematics* powerful. The *self-correcting coding theory* is one of the offspring of *Applied Finite Mathematics* and *Abstract Algebra*.

This is a compilation of lecture notes from a course for the second year graduate students at Purdue University. The lecture notes has four parts: Part **I** is about the elementary theory of coding. Part **II** is about algebraic coding. Part **III** is a survey of the useful part of algebraic geometry. Part **IV** is about geometric coding theory.

The ultimate goal of this text is to introduce the students to the modern *geometric Goppa codes* which is more accurate than the polynomial codes used in present day industries. However, the decoding process is slower. It called for more reseaches. Since the elementary coding theory is of interest only to students in engineering or computer science, hence unknown to many students in mathematics, we will begin with some discussions of this elementary material. It is naturally progressed to the algebraic codes, mainly BCH codes, Reed-Solomon codes and classical Goppa codes, which can be easily discussed in the context of linear algebra, polynomial rings and formal powerseries rings. A complete treatment of decoding polynomial codes is given in Part **II**.

One of the challenges is to present a survey of the useful parts of algebraic geometry which is an elegant and important topic in mathematics. Usually it takes several years intensive study for a student to master the subject. It is impossible to cover very much of the materials in one semester. Furthermore there are many ways to teach the subject. In the chapter on algebraic geometry, Part **III**, we adopt Chevalley's algebraic approach to the curve theory, and provide many examples to illustrate the theorems with many of the proofs being consigned to classic books by Zariski-Samuel [14], Chevalley [8] and Mumford [11].

The final part of this text is devoted to the well-known geometric Goppa codes [22] which are introduced in the standard fashion. Their early decoding processes depend on linear algebra only, and are elementary. For the decoding processes, we give explicit descriptions and try to present them as naturally as possible. Further discussion involves the remarkable concept of Feng and Rao [19] about majority voting will be treated in this manner.

The field of coding theory is too rich to be covered in a one semester course. We add an appendix to discuss the familiar topics as *convolutional codes, sphere packing problem,* other interesting *codes and Berlekamp Algorithm.* The appendix might be beneficial to interested readers who wish to have a wide scope of understanding the related materials.

This book is written to be short. There are about 155 pages for a one semester course. We hope that it will be useful to the students and working algebraic geometors alike in understanding the booming field of coding. We would like to thank B.Lucier for commenting on Parts I & II, W. Heinzer, D. Arapura, K. Matsuki for reading the part III on algebraic geometry and making some valuable suggestions, and N. Moses for reading and editing our manuscript.

Contents

INTRODUCTION

All living beings use signals to communicate with each other. The signals have forms of chemicals, sounds, colors etc. The signals are codes. About 2 million years ago, humanity departed from the apes by walking upright and creating abstract signals, which are mainly languages. All languages are codes. Many historians try to decipher "lost" languages. The most famous one was probably written hieroglyphs, which were deciphered using the Rosetta Stone. Since ancient times, many long poems have been passed down as an oral tradition, including *Illiad*, *Odyssey* in Greek or
it Mahabharata, *Ramayana* in Sanskrit or *Ode* in Chinese. One advantage of the rhyme and the poetic structure of poetic verse over prose is that it is easy to find the errors if they occur. In other words, poem is the first "error-detecting" form of communication.

In 1945, Erwin Schrödinger published a book entitiled *What is life? The Physical Aspect of the Living Cell*, in which he observed that chromosomes are code-scripts and are molecules in nature. There must be a code of some kind which allowed molecules in a cell to carry information. It motivated many scientists to study the codes transmitted by living beings, eventually leading to the discovery of the double helix structures of DNA and RNA by James D. Watson and Francis Crick.

In the meantime, a computer scientist, R.W. Hamming, used a primitive (by today's standard) computer to perform his research. At that time one had to queue one's work for the computer to a sequentially process. If the computer found errors in the program, it would skip the task and precede to the next one in the queue. The researcher would have to correct the errors in the faulty program, resubmit it to the queue, and wait several weeks for the computer to find time to work on it again. Apparently, Hamming was annoyed by the waiting and decided to create a "self-correcting code" which eventually bore his name, as follows:

Let us consider GF(2) (=$\mathbf{F}_2$), the field of two elements $\{0, 1\}$ which will be called the set of *alphabet* , and message $a_1a_2a_3a_4$ where $a_i \in \{0, 1\}$. Hamming added three more symbols $b_1b_2b_3$ by the following formula,

$$\begin{aligned} b_1 &= a_1 + a_3 + a_4 \\ b_2 &= a_1 + a_2 + a_3 \\ b_3 &= a_2 + a_3 + a_4 \end{aligned} \tag{1}$$

Then he used 7 symbols $a_1a_2a_3a_4b_1b_2b_3$ to carry the message of 4 symbols $a_1a_2a_3a_4$. We may consider the following matrix multiplication, with

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

and $[a_1a_2a_3a_4] \times G = [a_1a_2a_3a_4b_1b_2b_3]$. The matrix G is called the *generator matrix.* The a_i's are called the *message symbols.* Furthermore, let

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

and $[a_1a_2a_3a_4] \times G \times H = [000]$. The matrix H is called the *check matrix* and b_i's are called the *check symbols*.

The decoding process is as follows. Suppose that the computer, for whatever reasons, reads $[a_1a_2a_3a_4b_1b_2b_3]$ as $[a'_1a'_2a'_3a'_4b'_1b'_2b'_3]$ which might be different from the original string. However this kind of error is infrequent, so we may reasonably assume that there is at most one error, i.e., either

$$[a'_1a'_2a'_3a'_4b'_1b'_2b'_3] = [a_1a_2a_3a_4b_1b_2b_3].$$

or

$$[a'_1a'_2a'_3a'_4b'_1b'_2b'_3] = [a_1a_2a_3a_4b_1b_2b_3] + [0\cdots010\cdots0].$$

The computer calculates

$$[a'_1a'_2a'_3a'_4b'_1b'_2b'_3] \times H = [c_1c_2c_3].$$

If $[c_1c_2c_3] = [000]$, then the above defining equations (1) for b_1, b_2, b_3 show that there is either no error or there are more than two errors. Since we assumed that there is at most one error, we may conclude there is no error, so the computer should take the message $[a'_1a'_2a'_3a'_4]$. If $[c_1c_2c_3] \neq [000]$, then we have

$$\begin{aligned} &[a'_1a'_2a'_3a'_4b'_1b'_2b'_3] \times H \\ &= ([a_1a_2a_3a_4b_1b_2b_3] + [0\cdots010\cdots0]) \times H \\ &= [0\cdots010\cdots0] \times H \\ &= [c_1c_2c_3]. \end{aligned}$$

Therefore $[c_1c_2c_3]$ must be one of the row vectors of the matrix H, and thus the computer locates the position of the error. The computer simply flips the bit at that position. In this way the computer will correct the code $[a'_1a'_2a'_3a'_4b'_1b'_2b'_3]$ and take the first 4 bits of the corrected message as the message. This code not only detects the error but also corrects the error. Note that the *location* of the error must be detected before it can be corrected, a process that will be referred to as *error-locator*.

However, there might be 2 or more errors, in which case Hamming code fails and the above method decodes the message to the wrong word. One may assume that the possibility is rather small. The usage of Hamming codes is to eradicate a single error, but it is ineffective if the errors are multiple.

For a longer message, we can chop it into blocks(each with four bits), padding the end if necessary. This produces a *block* code.

The principles of self-correcting in Hamming codes are valid today, and widely used in communications through noisy channels. Academically, **code** means **self-correcting code**.

All channels of communications are noisy to different degrees. The self-correcting codes have become prominant today.

Although the above Hamming code is merely a simple exercise in linear algebra, the implications were shocking. At that time, C.E. Shannon was developing his theory of information. How do we measure *information*? Shannon took the idea of *entropy* from thermodynamics which is defined to be *logarithm of disorder*. The second law of thermodynamics states: *the entropy of the universe is always increasing*, simultaneously explaining the phenomena of cooling down, diffusion and the direction of time. Shannon defines

$$information = -entropy$$

Classically, given a discrete random variable X, i.e., one whose range $R = (x_1, x_2, \cdots)$ is finite or countable. Let $p_i = P(X = x_i) =$ the probability for x_i to happen. The *entropy* of X is defined by

$$H(X) = \sum_{p_i > 0} p_i log_2 \frac{1}{p_i} = -\sum_{p_i > 0} p_i \log_2(p_i).$$

Let us take a simple example. Suppose we check one out of two boxes (e.g. male and female) to provide some information. Before choosing a box, both boxes have equal chance of being selected, and we only pick one. So before the selection, the entropy is

$$\frac{1}{2} log_2 2 + \frac{1}{2} log_2 2 = 1 = -information.$$

After the selection, the entropy is

$$-1 log_2 1 = 0 = -information$$

The gain of information is

$$\text{gain of information} = 1$$

For 4 boxes to be checked, we either use the definition of entropy directly, or we may group them into two subsets; $\{$box 1, box 2$\}$ and $\{$box 3, box 4 $\}$, then we pick one subset out of the two subsets, and one box from the subset of two boxes, so

$$\text{gain of information} = 1 + 1 = 2$$

An ancient Chinese philosopher Laotzu said: "The way of nature is to decrease the excess and to increase the shortage, the way of human is to increase the excess and to decrease the shortage." A modern way is to say: "The way of nature is to increase entropy, the way of living being is to increase information (i.e., decrease entropy)". According to the second law of thermodynamics, the amount of entropy will increase or the amount of information will decrease as it passes though noisy channel. Hamming code is a way of protecting parts of the information so they decrease less with the help of a proper decoding procedure. The meaning of Hamming's discovery can be better understood using the framework of Shannon's information theory. Hamming's discovery led to BCH codes, Reed-Solomon codes (which are widely used today) and Goppa codes. Just like a tall tree starts from a tiny seed, the study of self-correcting codes eventually leads to many interesting problems for algebraic geometors.

Let us return to the codes of life, DNA and RNA. Today they are broken down into basic units called *genes*. There is an important phenomenon called *mutation* which is caused by the environmental cosmic rays (lately, biologists add *free radicals* as other mutation agents) and is a fundamental force of evolution. The cosmic rays will input energies and change the configurations of the molecules of the genes. The mutations can be considered as errors. So there are errors in the genes transmitted from generations to generations. Most mutations are detrimental and are not welcomed by living beings. The natural selection of evolution will weed out the bad mutations. In any case, the mutations are most likely wastes for the advanced living beings. How to deal with them? There may be some other machinaries, say a self-correcting function, to prevent those mutations from happening. We find out that some genes are functional and some are *non-coding genes* or *junk genes* and genetic engineers typically transplant only the functional genes to other species. We know that life is very economical, so why is it that the non-coding genes are passed down through the generations? Beside many functions studied by genealogists, maybe they are for a self-correcting purpose just like the additional symbols $b_1b_2b_3$ in the our example of Hamming code. It turns out that DNA has some *proof-reading* capabilities which RNA lacks, although scientifically how those *proof-reading* capabilities function is unclear. However, these capabilities slow down the rates of mutations considerably. These lackings of proof-reading capabilities for RNA make some viruses evolve very fast and cause many problems for health care. These self-correcting codes that occur in nature might be better than all of our coding theory which are based on algebra or algebraic geometry. It is a myth in biology.

Similar to the codes of life, our civilizations and cultures themselves may be viewed as the transmissions of codes through time. Due to the nature of decaying phenomena caused by historical events, thermodynamics, and cosmic rays, we may view the channel of time as an noisy channel. In his old age, Leonado da Vinci worried about the decaying of his masterpieces. How to preserve our heritages becomes an important topic. One way might be using the self-correcting codes to prolong the useful period of our civilizations and cultures. We confess our ignorance on this subject, The oral and written languages are important parts of heritages. In all oral and written languages, there are many *non-functional* parts which serve as check symbols. It might be dangerous to delete those parts.

Today is an information era. Messages are transmitted in sequences of 0,1's through space. It is possible to make an error with the noisy channels, and so self-correcting codes become our saviors to eradicate all errors as long as the numbers of errors are small. Self-correcting codes are widely used in industries for a variety of applications including e-mail, telephone, remote sensing (e.g. Mars photographs), CD etc. We will present some essences of the theory in this book.

Using linear algebra, we have Hamming codes. The next level of coding theory is through the usage of polynomials and power series produces BCH codes, Reed-Solomon codes and classical Goppa codes. The more advanced level of coding theory is an application of algebraic geometry which is the geometric Goppa codes. The aim of this book is to bring interested readers gradually to the highest level of coding theory.

PART I

ELEMENTARY CODING THEORY

Chapter 1

Linear Codes

In this chapter, we will lay a foundation for coding theory using linear algebra, although certainly there are many other ways. For instance, we could simply send multiple copies of the same message and determine the correct one by a majority vote which is called *repetition* code. Alternatively, we could use curve theory to construct a "self-correcting code" as follows.

Let us consider all quadratic curves defined by equations

$$y = f(x) = a_0x^2 + a_1x + a_2$$

representing the original data $\{a_0, a_1, a_2\}$. We transmit the values $y_0 = f(0), y_1 = f(1), y_2 = f(2)$, instead of $\{a_0, a_1, a_2\}$, making the observation that $\{a_0, a_1, a_2\}$, and $\{y_0, y_1, y_2\}$ determine each other. To correct one possible error, we should add two more symbols $y_3 = f(3), y_4 = f(4)$ and transmit $\{y_0, y_1, y_2, y_3, y_4\}$, because any four correct values will determine the quadratic curve, and hence the remainning one. That is, a brute-force search for the correct tuple will reveal which four values are consistent (for a quadratic curve).

So the process is simple; Take any original message $\{a_0, a_1, a_2\}$ and determine five values $\{y_0, y_1, y_2, y_3, y_4\}$, where we have each $y_i = f(i) = a_0i^2 + a_1i + a_2$. Upon transmission of the five values $\{y_0, y_1, y_2, y_3, y_4\}$, suppose the other end, the receiver receives $\{y_0', y_1', y_2', y_3', y_4'\}$ with one possible error. The receiver then determines which of the four tuples are consistent (for a quadratic curve), and then use it to correct the fifth one. In general, when we consider curves of higher degrees and we may transmit sufficiently more points than necessary, a brute-force search for the correct tuple will be time-consuming. However, we may modify this curve code slightly to produce the Reed-Solomon code (see below) with a fast decoding process.

Both *repetition* code and curve codes are time-consuming. Instead we will focus on *linear code*, which is more efficient and has a fast decoding method. We will follow the historical development of the theory of self-correcting codes, primarily using techniques from linear algebra.

1.1 Preliminaries

Groups and Fields

The abstract *group theory* was an invention of E. Galois. It turns out to be an important concept of mathematics, physics etc. Let us give the usual definition of *Group*,

Definition : Let G be a set. A binary operation $\cdot$ is a rule to assign an element $c = a \cdot b$, given any pair of elements (a, b) in G. If $c = a \cdot b \in G$ always, then we say that G is *closed* under the binary operation $\cdot$. If G is closed under $\cdot$ and has the following three properties, then we say G forms a *group* under the binary operation $\cdot$

(1) *Associative law* : $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.

(2) *The existence of identity* : there exists an element e such that$a \cdot e = e \cdot a = a$.

(3) *The existence of inverse* : for any a, there exists an element b such that $a \cdot b = b \cdot a = e$.

Sometimes we omit the mention of $\cdot$ if it is obvious, and we simply say that G is a group. If $a \cdot b = b \cdot a$ always, we say that G is an *abelian group*. ■

Usually we work on a set of symbols. We may put rules on operations on the set of symbols to make them mathematically easy. We may impose two operations, i.e., *summation* and *multiplication*. We shall consider the rules satisfied by them. In the primary school and the high school we have studied the *associative law*, the *commutative law*, the *distributive law* and the additive identity $0 \neq 1$ the multiplicative identity for integers Z, now we will assume that the same rules are satisfied by the two imposed operations, summation and multiplication, on the set of symbols, furthermore, we will include two further useful rules which are valid for the set of rational numbers Q, i.e., (1) every non-zero element will have an inverse, and (2) if $ab = 0$, then one of a, b must be zero. for our set. We thus called a non-empty set K with two operations satisfies all the preceding requirements a *field*. So algebraically a *field* behaves as the rational numbers and serves as the set of numerals in all our computations.
Example 1: The well-known rational numbers Q, real numbers R and the complex numbers C are all fields. There are other important fields Z/pZ, the field of the equivalent classes $[a]_p$ modulus a prime number p,

$$[a]_p = \{\text{all integers b such that } p \mid a - b\}.$$

where p is a prime integer and the summation, product are defined as

$$[a]_p + [b]_p = [a + b]_p,\ [a]_p \cdot [b]_p = [ab]_p$$

■

It is routine to show that $Z_p = Z/pZ$ is a field if p is a prime number (see the exercise). We have the following **Proposition-Definition**,

Proposition 1.1.1 *: Let* $\mathbf{K}$ *be a field. Consider the set* $S = \{1, 1+1, \cdots, \overbrace{1 + 1 + \cdots + 1}^{n}, \cdots\}$ *where* 1 *is the multiplicative identity in the field* K*. Then either the the set* S *does not contain*

0 or the first n such that $n \cdot 1 = 0$ must be a prime number p. In the first case, we say the field K is with characteristic 0 and in the second case we say the field is with characteristic p.

Proof. The only thing we have to prove is that if $n\dot{1} = 0$, then n must be a prime number. It follows from

$$n \cdot m \cdot 1 = (n \cdot 1)(m \cdot 1)$$

■

Vector Spaces

Before we go further, we need some elementary knowledge of *Linear Algebra*. Let us defining the concept of *vector space*,

Definition : Let K be a field and V be a non-empty set. A *vector space* $(V, +, \cdot, K)$, where $+$ is a binary operation between elements in V and $\cdot : K \times V \mapsto V$ is a binary operation, satisfies the following conditions,

(1) $(V, \cdot)$ is an abelian group.
(2) *Associative law* : $a \cdot (b \cdot v) = (a \cdot b) \cdot v$, *for all* a, b, v.
(3) *Distributive law* : $(a + b) \cdot v = a \cdot v + b \cdot v$ and $a \cdot (v + u) = a \cdot v + a \cdot u$.
(4) $1 \cdot v = v$

sometimes we omit the mention of $\cdot$ if it is obvious. ■

If we have two fields $L \supset K$, then clearly L is a vector space over K.

Finite Fields

We need some basic knowledge of the field theory. The reader is refered to *Commutative Algebra*, Vol I, page 60 [14] by Zariski-Samuel, for it and the following Corollary,

Corollary 1.1.2 : *If L is an overfield of K and $x \in L$, then x is algebraic over K if and only if $K(x)$ is a finite extension of K. In that case, if $n = [K(x) : K]$ (i.e., $n =$ the vector space dimension of $K(x)$ over K), then the minimal polynomial of x in $K[X]$ is n.*

Proof. See the reference. ■

We define the *prime field* as Z/pZ or the rational number field Q. In every field K there must be a copy of $Z/pZ = Z_p$ for some p or Q which determines its charateristic. The definition of *algebraically closed field* is common. The reader is refered to *Commutative Algebra*, Vol I, page 106 [14] by Zariski-Samuel, for it and the following Theorem,

Theorem 1.1.3 : *If K is a field, then there exists an algebraic closure of K and any two algebraic closure of K are K-isomerphic fields.*

Proof. See the reference. ∎

In the *coding theory*, we are interested in a finite field L. Naturally it contains some prime field Z_p which is with exactly p elements. Therefore L is a finite dimensional vector space over Z_p. By the vector space consideration of dimension, let the dimension by n, then put a coordinate system to it, we conclude that it has precisely p^n elements, i.e., the following theorem

Theorem 1.1.4 *: If K is a finite field, then there are exactly p^n elements for some suitable prime number p and a positive integer n.*

Proof. See the above discussions. ∎

In coding theory, sometimes we consider a finite field F which is a field between a prime field Z_p and a fixed algebraic closure Ω of F.

1.2 Linear Codes

In general, fix an algebraic closure Ω of Z_p, we will consider a finite field K of $q = p^m$ elements between Ω and Z_p. The field is uniquely detremined by the number q, and will be denoted by $\mathbf{F}_q$. The detailed treatments of finite fields was given in a later section.

Let us refer to the finite field $\mathbf{F}_q$ as our set of *alphabets*, then any message $a_1a_2\cdots a_k$ can be considered as a vector $[a_1a_2\cdots a_k]$ in the vector space $\mathbf{F}_q^k$. We will not consider any basis except the the standard basis in $\mathbf{F}_q^k$, and the vector space $\mathbf{F}_q^k$ with the *standard basis* will be called the *message space* $\mathbf{V}$.

The system of equations (1) in the introduction defines a subspace $\mathbf{C}$ in $\mathbf{F_q^n}(= \mathbf{U})$ where $n > k$. Any element in $\mathbf{C}$ will be called a *code word*, and any element in $\mathbf{F}_q^n(= \mathbf{U})$ will be called a *word*. The subspace $\mathbf{C}$ will be referred to as the *code space*.

Definition : A $[n, k]$ code is an k-dimensional subspace $\mathbf{C}$ of an n- dimensional vector space $\mathbf{F}_q^n(= \mathbf{U})$ with standard basis. Sometimes, the number k will be referred as the *rank* of $\mathbf{C}$. A *coding* is a $1-1$ (i.e., injective) map from the message space $\mathbf{V}$ to the code space $\mathbf{C}$.

All concepts and theorems in coding theory are independent of the ordering of the basis vectors, so we will allow permutations of the standard bases in the message spaces and word spaces. Two $[n, k]$ codes $\mathbf{C_1}, \mathbf{C_2}$ are said to be *equivalent* if their code spaces U_1, U_2 differ by permutations of the standard bases of the message space and word space. ∎

A k-dimensional subspace $\mathbf{C}$ is generally defined by $n - k$ equations. To apply Gaussian elimination to simplify the system, we are restricted to using row operations and permutations of columns only. Therefore we may assume that the system of equations is of the following *standard* form,

$$y_1 \quad = c_{1,1}x_1 + \cdots + c_{1,k}x_k$$

$$\cdots\cdots \qquad (2)$$
$$y_{n-k} = c_{n-k,1}x_1 + \cdots + c_{n-k,k}x_k.$$

It follows that the *generator matrix* G can be written as the following *standard* form,

$$G = \begin{pmatrix} 1 & 0 & \cdots & 0 & c_{1,1} & \cdots & c_{n-k,1} \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & 0 & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & 1 & c_{1,k} & \cdots & c_{n-k,k} \end{pmatrix}$$

with $[x_1 \cdots x_k] \times G = [x_1 \cdots x_k y_1 \cdots y_{n-k}]$. The symbols x_i's will be called *message symbols* and symbols y_i's will be called *check symbols*. Define G' by the equation (2) where

$$G = \begin{pmatrix} I & G' \end{pmatrix}$$

and I is the $k \times k$ identity matrix.

Observe that if we rewrite the system of equations (2) in the following form,

$$y_1 \quad - c_{1,1}x_1 - \cdots - c_{1,k}x_k = 0$$
$$\cdots\cdots \qquad (2')$$
$$y_{n-k} - c_{n-k,1}x_1 - \cdots - c_{n-k,k}x_k = 0,$$

then the *check matrix* H can be written as

$$H = \begin{pmatrix} -c_{1,1} & \cdots & -c_{n-k,1} \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ -c_{1,k} & \cdots & -c_{n-k,k} \\ 1 & \cdots & 0 \\ \cdots & \cdots & \cdots \\ 0 & \cdots & 1 \end{pmatrix}$$

It is easy to verify that $G \times H = 0$, hence we have $[x_1 \cdots x_k] \times G \times H = [0 \cdots 0]$. We may write

$$H = \begin{pmatrix} -G' \\ J \end{pmatrix}$$

where J is the $(n-k) \times (n-k)$ identity matrix.

For a longer message, we chop it into blocks, each having k elements from $\mathbf{K}$, padding the end if necessary and apply the above method for each block. This is so called *block* code.

We may identify the message space $\mathbf{V}$ with a subspace $\{[x_1 \cdots x_k 0 \cdots 0] : x_i \in \mathbf{F}_q\}$. If there is no confusion, we will call this subspace the message space $\mathbf{V}$.

Let us consider the coset space $\mathbf{U}/\mathbf{V}$, the image space of the mapping defined by H. It is called the *syndrome space*. It can be canonically identified with the subspace $\mathbf{W} = \{[0 \cdots 0 y_1 \cdots y_{n-k}] : y_j \in \mathbf{F}_q\}$. For any element $a \in \mathbf{U}$, $a \times H$ is called the *syndrome* of a. We have

Proposition 1.2.1 *: Let $a, b \in \mathbf{U}$. Then they have the same syndrome or they belong to the same coset iff $a \times H = b \times H$. In particular, a is in the code space iff $a \times H = 0$.*

■

Example 1: The example of Hamming in the introduction is a $[7,4]$ code. Its syndrome space $\mathbf{W}$ is $\{[0000y_1y_2y_3] : y_j \in \mathbf{F}_q\}$ Let us consider a general $[n,k]$ code $\mathbf{C}$. From the check matrix H, we may add $n \times k$ 0's to form the following $n \times n$ matrix $\bar{H}$,

$$\bar{H} = \begin{pmatrix} 0 & \cdots & 0 & -c_{1,1} & \cdots & -c_{n-k,1} \\ 0 & \cdots & 0 & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & -c_{k,1} & \cdots & -c_{n-k,k} \\ 0 & \cdots & 0 & 1 & \cdots & 0 \\ 0 & \cdots & 0 & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & 0 & \cdots & 1 \end{pmatrix}$$

Then it is easy to see that $\bar{H}^2 = \bar{H}$ and $\bar{H}$ is the projection of the whole space $\mathbf{U}$ to the syndrome space $\mathbf{W}$.

■

1.3 Distances

The only natural metric in $\mathbf{F}_q$ is the discrete one, i.e., we have $d(a,b) = 1$ if $a \neq b$ and $d(a,b) = 0$ if $a = b$. We generalize this distance function to n-dimensional vector space $\mathbf{F_q^n}(= \mathbf{U})$ by the following definitions,

Definition : Let $a = [a_1 \cdots a_n], b = [b_1 \cdots b_n] \in \mathbf{U}$. Then we define the *Hamming distance* between a, b as the product distance of the above mentioned natural metric in $\mathbf{F}_q$ (which may be called a sum distance), i.e., $d(a,b) = \sum d(a_i, b_i)$ where $d(a_i, b_i)$ is the natural distance of elements in the field $\mathbf{F}_q$. ■

Definition : Let $a = [a_1 \cdots a_n] \in \mathbf{U}$. Then we define the *Hamming weight* of a as $w(a) = d(a, 0)$. ■

We have the following natural theorem,

Theorem 1.3.1 *: The Hamming distance is a distance, i.e.,*

$$(1) : d(a,a) = 0.$$
$$(2) : d(a,b) = d(b,a).$$
$$(3) : d(a,b) + d(b,c) \geq d(a,c)$$

Proof. It is routine for all product distances. ■

Proposition 1.3.2 *: We always have $d(a,b) = w(a-b) = d(a-b, 0)$.*

Proof. It is evident. ■

Definition : The minimal distance of a set S is $min\{d(a,b) : a \neq b,\ a, b \in S\}$. Sometimes the minimal distance of a set S is called the *Hamming distance* of S. ■

1.4 Maximum Likehood Decoding

Suppose we have received a message $a' = [a'_1 \cdots a'_n]$, and we know the code word $a = [a_1 \cdots a_n]$ is in the code space $\mathbf{C}$ but perhaps $a' \notin \mathbf{C}$. How do we recover the code word a? Let us assume that $a' = a + e$ where e is the *error* vector. Since $a = a' - e$, a and e uniquely determine each other with a' known. However a' and e do not uniquely determine each other. We may select e from the syndrome space with $e \times \bar{H} = a' \times \bar{H}$, and let $a = a' - e$. Note that we have

$$a \times \bar{H} = (a' - e) \times \bar{H} = 0$$

Therefore we conclude that $a \in \mathbf{C}$ the code space. However, upon closer examination, we discover that we may use any element in the same coset as e. Up to now, there is no particular reason to pick any one from the syndrome space. We have to find some criterion for our selection.

A meaningful way of decoding is to consider the real situation. For the error vector e, a zero coordinate indicates that there is no error in that position, a non-zero coordinate indicates that the letter at that position should be corrected. Let p_i be the probability (independent of a') that the error vector e at a particular position has a value i. For simplicity we assume that the following definition is satisfied,

Definition (*symmetric channel*) If $p_i = p_j (= p)$ for all $i, j \neq 0$, and $p_0 > 1/2$, then it is said to be a *symmetric channel*. ■

Note that $\sum_{i=0}^{q-1} p_i = 1$, therefore $p = p_i < 1/(2(q-1))$ for all $i \neq 0$. Note that p_0 is the probability that there is no error at this particular position. In other words, the condition $p_0 > 1/2$ simply means that it is less likely to have an error. We shall assume that we have a *symmetric channel*.

Proposition 1.4.1 : *For a symmetric channel, the probability $P(e)$ for an error vector e to happen is $p^{w(e)}(1 - w(e)p)^{q-w(e)}$. Therefore $P(e) > P(e')$ iff $w(e) < w(e')$.*

Proof. It is evident. ■

The *maximum likelihood decoding* is to find an error vector e in the coset S of a' such that

$$w(e) = d(e, 0) = min\{w(b) :\ b \in\ S\}.$$

The previous Proposition explains the meaning of the term *maximum likelihood decoding*. We further define

Definition : A *coset leader* e of a coset S is an element e in S such that

$$w(e) = d(e, 0) = min\{w(b) :\ b \in\ S\}.$$

■

We have the following figure.

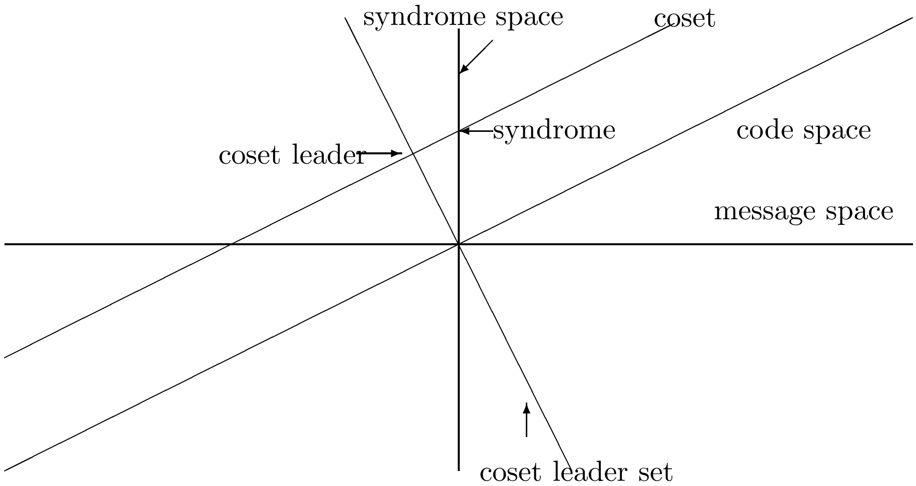


Figure 1

The above figure over the field of real numbers is misleading in the context of a finite field, where the coset leaders only forms a set rather than a vector subspace as in the real case. The coset leaders may not even be unique for one coset as indicated in the following example.

Example 2: Let us consider a $[6,3]$ code $\mathbf{C}$ over the algebraic set $F_2 = Z/2Z$with the following check matrix H,

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

Furthermore we let $\bar{H}$ be the following matrix,

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

If the coset S be with syndrome $[000111]$ i.e., $s \times \bar{H} = [000111]$ for any $s \in S$, then it is easy to see that no element s with $w(s) = 1$ is in S, while there are three elements

$[100001], [010100], [001010]$ in S. They are all coset leaders and they do not form a vector space. ■

If there are several coset leaders for a coset, we will identify any one of them to be the coset leader of the coset. The maximum likelihood decoding procedure is, for any message a', we find the coset S where it lies (usually by finding the syndrome of a' which determines the coset S), then we find a coset leader e which is a most likely error vector in S. Finally we correct the error by taking $a = a' - e$. In engineering, we may have a table of $\{(syndrome, coset\ leader)\}$ for the sake of convenience. Since the syndrome space is of dimension $n - k$ consists of q^{n-k} elements. If $n - k$ is small, we may precompute the table and decode accordingly.

Note that this particular decoding procedure may not be effective for all codes, so we shall look for other possible decoding procedures. Let us continue our discussions about possible codes.

1.5 Dual Codes

Let C be a linear code, i.e., C is vector subspace of $\mathbf{F}_q^n$. Let us introduce a *pairing* $< v \mid u >$ on $\mathbf{F}_q^n$ as follows.

Definition : Let $v = (v_1, \cdots, v_n), u = (u_1, \cdots, u_n)$ be two vectors $\in \mathbf{F}_q^n$ with respect to a basis. Note that we only allow a basis to be a permutation of the standard basis. We define

$$< v \mid u >= \sum_{i=1}^{n} v_i u_i$$

■

We define

Definition : Let $C^{\perp} = \{v \in F_q^n :< v \mid u >= 0\ for\ all\ u \in C\}$. If C is a linear code, then $C^{\perp}$ is called the *dual code* of C. ■

Proposition 1.5.1 *: Given C a linear code, its dual $C^{\perp}$ is a linear code. If C is a $[n, k]$ code, then $C^{\perp}$ is a $[n, n-k]$ code.*

Proof. Left to the reader. ■

Exercises

(1). Prove that $\mathbf{Z}/p\mathbf{Z}$ is a field, while $\mathbf{Z}/p^m\mathbf{Z}$ is not a field for $m \geq 2$.

(2). Let us consider the repetition code $[a_1 \cdots a_n] :\mapsto [a_1 a_1 \cdots a_1 \cdots a_n a_n \cdots a_n]$ where each digit repeats itself m times. Find the generator matrix, check matrix and the minimal distance of this code.

(3). Prove that for a given binary $[n, k]$ code with at least one word of odd weight, then all code words of even weight form an $[n, k-1]$ code.

(4). Let us consider the example of Exercise (2). If we want to correct 2 errors, how long should be the code? If we want to correct ℓ errors, how long should be the code?

(5) If C is a $[n, k]$ code, then $C^{\perp}$ is a $[n, n-k]$ code.

(6) Let $\mathbf{C}$ be a code space. Show that

$$min\{d(a,b) : a, b \in \mathbf{C}\} = min\{w(a) : a \in \mathbf{C}\}$$

(7) Let C be the *repetition* code with $n = 3 = m$ (cf Exercise (2)). What is $C^{\perp}$?

1.6 Hamming Code

Let us consider the case of *binary code*, i.e., $q = 2$, $\mathbf{F}_q$=$\mathbf{F}_2$. For $n \geq 3$, a *Hamming code* is a code with the check matrix H where the rows of H are all non-zero elements of $\mathbf{F}_2^n$. Note that the number of rows is $2^n - 1$ and H is a $(n) \times (2^n - 1)$ matrix, and the code thus constructed is a $[2^n - 1, 2^n - n - 1]$ code. After a suitable permutation of rows, we may assume that the last n rows form an identity matrix J and hence H is in *standard form*.

For the purpose of decoding, we may arrange the rows of H differently. Recall the *binary expansion* of an integer i between 1 and $2^n - 1$ as follows;

$$i = \sum_{j=1}^{n} a_j 2^{j-1}, \qquad 1 \leq i \leq 2^n - 1, \quad a_j \in \mathbf{F}_2$$

and we put $[a_1 \cdots a_n]$ as the i-th row of H. Now suppose that a word $[b_1' \cdots b_{2^n-1}']$ has at most one error. In other words, if $[b_1 \cdots b_{2^n-1}]$ is the correct word, then either

$$[b_1' \cdots b_{2^n-1}'] = [b_1 \cdots b_{2^n-1}]$$

or

$$[b_1' \cdots b_{2^n-1}'] = [b_1 \cdots b_{2^n-1}] + [0 \cdots 010 \cdots 0].$$

The computer calculates

$$[b_1' \cdots b_{2^n-1}'] \times H = [c_1 \cdots c_n]$$

If $[c_1 \cdots c_n] = [0 \cdots 0]$, then we can see there is no error because if there is a single error, we must have

$$[b_1' \cdots b_{2^n-1}'] \times H = ([b_1 \cdots b_{2^n-1}] + [0 \cdots 010 \cdots 0]) \times H = [0 \cdots 010 \cdots 0] \times H \neq [0 \cdots 0].$$

A contradiction. If $[c_1 \cdots c_n] \neq [0 \cdots 0]$, then there must be an error, and by the above computation it is easy to see that the error appears at the i-th position where

$$i = \sum_{j=1}^{n} c_j 2^{j-1}.$$

The method is effective in both locating and correcting the error, since error correction here is simply switching 0 and 1. The shortcoming is that the Hamming codes are unable to correct several errors (in particular a burst of errors).

This calls for more theoretical study of Hamming codes. We have

Proposition 1.6.1 *: The minimal distance of all pairs of elements of any Hamming code* $\mathbf{C}$ *is three. In other words,*

$$min(w(a)) = 3, \ for \ all \ a \neq 0 \ in \ \mathbf{C}.$$

Proof. We have $a \in C \Leftrightarrow a \times H = 0$. This equation can be interpreted a linear equation among the rows of H. Since all row vectors of H are non-zeroes, we have $min(w(a)) > 1$. Since any two row vectors are linearly independent, we have $min(w(a)) > 2$. We can easily find three row vectors which are linearly dependent, so we conclude $min(w(a)) = 3$. ∎

The concept of the above proposition is very important. so we have the following definition.

Definition : Let $\mathbf{C}$ be an $[n, k]$ code. If

$$d = min(w(a)), \ for \ all \ a \neq 0 \ in \ \mathbf{C}$$

Then $\mathbf{C}$ will be called an $[n, k, d]$ code. ∎

The following proposition was discovered by R. Singleton in 1964.

Proposition 1.6.2 *(Singleton Bound): Let* $\mathbf{C}$ *be an* $[n, k, d]$ *code. Then* $d \leq n - k + 1$.

Proof. Let π be the projection of the word space to the last $n - d + 1$ coordinates. Let us restrict π to the code space $\mathbf{C}$. Let two code words $c \neq c'$, or $c - c' \neq 0$. Then there must be at least d non-zero coordinates for $c - c'$. Clearly $\pi(c - c') \neq 0$ or $\pi(c) \neq \pi(c')$. Therefore the restriction of π is a one-to-one map on $\mathbf{C}$. Look at the dimensions, we conclude that k=dim ($\mathbf{C}$)$\leq n - d + 1$ **and** $d \leq n - k + 1$**.** ∎

Definition : A code that satisfies the Singleton bound with equality is called an *MDS* code (maximum distance separable code). ∎

Note that all Hamming codes are $[2^n - 1, 2^n - n - 1, 3]$ codes for $n \geq 3$. None of them is a *MDS*code. We have the following proposition,

Proposition 1.6.3 *: Let* $\mathbf{C}$ *be an* $[n, k, d]$ *code. Then for any word* $a' = [a'_1 \cdots a'_n]$ *with less than or equal to* $\lfloor (d-1)/2 \rfloor$ *errors, there is at most an unique element* $a = [a_1, \cdots a_n] \in \mathbf{C}$ *such that* $d(a, a') \leq \lfloor (d-1)/2 \rfloor$.

Proof. It follows from the statement that there is a code word a, such that $d(a, a') \leq \lfloor (d-1)/2 \rfloor$. If there are two elements $a = [a_1, \cdots a_n]$, $b = [b_1, \cdots b_n] \in \mathbf{C}$ which satisfy the criteria of the Proposition, then $d(a, b) < d$. A contradiction. ∎

The above proposition states that theoretically the code $\mathbf{C}$ may correct up to $\lfloor (d-1)/2 \rfloor$ errors. It follows that Hamming code may correct 1 error which we already know. Later, we will construct $[n, k, d]$ code for large d.

Another important property of Hamming codes $\mathbf{C}$ is that any word $a' = [a'_1 \cdots a'_n]$ is within a distance of 1 of the code space, because $a' \times H = [c_1, \cdots, c_n] = [0 \cdots 010 \cdots 0] \times H$, therefore $(a' + e) \times H = [0 \cdots 0]$, where $e = [0 \cdots 010 \cdots 0]$, and $a' + e \in \mathbf{C}$. We define

Definition : Let $\mathbf{C}$ be an $[n, k, d]$ code. If all words are within a distance of $\lfloor (d-1)/2 \rfloor$ of $\mathbf{C}$, then $\mathbf{C}$ will be called a *perfect code.* ■

Remark: A perfect code $[n, k, d]$ can be theoretically decoded in an unique way if only $\lfloor (d-1)/2 \rfloor$ errors are allowed. In other word, if there are more than $\lfloor (d-1)/2 \rfloor$ errors, it might be decoded to a wrong word.

Let us compute the probability for a decoder which can correct t errors on a $[n, k]$ code to fail. Note that for Hamming codes, $t = 1$. If there are $t + 1$ or more errors, then the decoder will not decode correctly. Let the channel have a probability p of being incorrect and q of being correct. Then $p + q = 1$ and

$$1 = 1^n = (p+q)^n = \sum_{i=0}^{n} \mathbf{C}_i^n p^i q^{n-i}$$

the probability of failing to decode or decoding improperly is

$$r = \sum_{i=t+1}^{n} \mathbf{C}_i^n p^i q^{n-i} = 1 - \sum_{i=0}^{t} \mathbf{C}_i^n p^i q^{n-i}$$

■

Exercises

(1): Show that a Hamming code is a perfect code (note that in the definition of Hamming code, we assume that $n \geq 3$).

(2): Show that a *repetition code* of odd length $2e + 1$ i.e. a code with generating matrix $[11 \cdots 11]$ is a perfect code.

(3): Let C be a binary perfect code of length n with minimum distance 7. Show that $n = 7$ or $n = 23$.

(4): Let p be a prime number and $q = p^m$. A q-ary Hamming code of length $(q^n - 1)/(q - 1)$ is defined by a check matrix H with the properties that (1): the zero vector is not any row vector. (2): any two row vectors are linearly independent. (3): any non-zero vector is linearly dependent on one of the row vectors. Show that it is a perfect code.

(5): Set up a computer program to decode a Hamming code.

(6): Show that if there are more than 2 errors in a received word r for a Hamming code, than r will be decoded to a wrong code word.

1.7 Shannon's Theorem

For simplicity, we assume that $q = 2$ in this section. For all different codes we need a common standards of measurements. We have

Definition : The *rate of information* R of an $[n,k]$ code is defined to be k/n. ■

We will consider all codes (linear or not). A *code* is defined as a subset $M = \{a_1, \cdots, a_m\}$, with m elements, of $\mathbf{F}_q^n$=$\mathbf{U}$. We further assume that all a_i's occur with equal probability. The maximun likelihood decoding is understood to decode any received word a' as $a \in M$ with $d(a,a')$ minimal. We shall generalize the above definition to the general cases. For the linear code space, the number of elements is q^k, and $k/n = log_q(q^k)/n$. Therefore we naturally generalized the above definition to the following,

Definition The *rate of information* R of a code M of m elements in F_q^n is defined to be

$$R(M) = n^{-1}\ log_q\ m.$$

■

Let P_i be the probability of incorrect decoding for $a_i \in M$. Then

$$P(M) = m^{-1} \sum P_i$$

is defined to be the average probability of incorrect decoding for the code M.

Definition : The *rate of distance* of an $[n,k,d]$ code is defined to be $\delta = d/n$. ■

Proposition 1.7.1 *: The rate of information of an* $[n,k,d]$ *code at most* $1 + 1/n - \delta$*.*

Proof. It follows from **Proposition 1.6.2**. ■

It follows from the preceding **Proposition** that as the rate of distance δ is larger, the rate of information is smaller. We are working towards the statement of Shannon's Theorem [30], which has been very influential to the development of coding theory. We will consider all codes (linear or not).

Suppose that the message is transmitted through a *binary symmetric channel*, i.e., the symbols 0, 1 have equal probability $\wp$ of be trasmited incorrectly. The value of $\wp$ is $< 1/2$. We shall have a common measurement of all channels. There are only two possible outcomes, either correct or incorrect, the classical entropy function $\wp log_2 \wp + (1-\wp) log_2(1-\wp)$ gives a good measure. Except its values are not ideal, so add 1 to the function.

We define

Definition : The *capacity* of the transmission is

$$c(\wp) = 1 + \wp\ log_2\ \wp + (1-\wp) log_2\ (1-\wp)$$

where $\wp < 1/2$. ■

The capacity is 1 if $\wp = 0$, and close to 0 if $\wp$ is close to $1/2$. We have the following proposition,

Proposition 1.7.2 *The capacity is a monotonic decreasing function of* $\wp$ *from* 0 *to* 1/2*.*

Proof. Let $c(x)$ be the capacity function. Then we have

$$c'(x) = log_2\ x - log_2(1-x) = log_2 \frac{x}{1-x}$$

which is negative as long as $0 < x < 1/2$, hence $c(x)$ is monotonic decreasing between 0 and 1/2.

■

Observe that the above definitions are consistent with all of the previous definitions given in the context of linear codes. Now we may state *Shannon's Theorem* [30];

Theorem 1.7.3 *(Shannon's Theorem): For any rate of information $0 < R <$ capacity, there is a sequence of codes $\{M_n\}$ such that $R(M_n) > R$ and the average probability of incorrect decoding $P(M_n) \to 0$ as $n \to \infty$.*

■

We will skip the proof of the above existence theorem. Although it is interesting conceptually, there is no constructive proof of Shannon's Theorem. In the light of the above Shannon's Theorem, observe that the $[2^n - 1, 2^n - n - 1, 3]$ Hamming codes $\mathbf{C_n}$ will have

$$R(\mathbf{C_n}) \to 1$$
$$\delta = 3/(n^2 - 1) \to 0,$$

i.e., the correcting power tends to zero as $n \to \infty$, while the above theorem states that the best sequence $\{M_n\}$ will have the oppositing sequeence $P(M_n) \to 0$. In other words, Hamming codes are not the best that we should expect. We may wish to construct other sequence of linear codes that do fulfill some of the expectations. It gives a powerful incentive to search for long codes, and it separates the study of codes to two branches: theoretical codes disregarding the decoding procedures and the decoding procedures themselves. Certainly for practical applications, we can not allow $n \to \infty$. The interested reader of Shannon's theorem is referred to [J.H.van Lint [7]].

1.8 Gilbert-Varshamov's Bound

Let us consider the n dimensional vector space $\mathbf{V}$=$\mathbf{K}^n$ over a finite field $\mathbf{K}$ of q elements. Furthermore let vector $\alpha \in \mathbf{V}$ and $\mathbf{B}(\alpha$, r) be the ball of radius $r \leq n$ arround α, i.e.,

$$\mathbf{B}(\alpha, r) = \{\beta : d(\alpha, \beta) \leq r\}$$

Define its volume $\mathbf{V}(\alpha$, r) to be the number of elements inside $\mathbf{B}(\alpha, r)$. We have the following geometric proposition,

Proposition 1.8.1 *We have $\mathbf{V}(\alpha,\ r) = \sum_{i=0}^{r} \mathbf{C}_i^n (q-1)^i$. Note that it is independent of α, and we will denote the common number by $\mathbf{V}_q(n,\ r)$.*

Proof. Let $0 < d(\alpha, \beta)$=i $\leq$ r. Then α, β differ at i positions. There are $\mathbf{C}_i^n$ selections of the positions and there are $(q-1)^i$ possible values for those different positions and hence the proposition.

■

Let us consider a code $\mathbf{M} \subset F_q^n$ which may not be linear, and let m be the number of elements in $\mathbf{M}$. We will say this is an (n, m, d) code where n is the dimension of the word space and d is the minimal Hamming distance between elements of $\mathbf{M}$. For a fixed pair (n, d), we may consider the maximal possible m as follows,

Definition : Let A(n, d)=Max$\{m \mid$ an (n, m, d) code exists. $\}$. ■

For a sequence of (n_i, m_i, d_i) codes, we may assume that they have the same rate of distance δ, i.e., $d_i = n_i\delta$. Then we may consider **the limit of the rate of information of the seqence** be $lim_{i \to \infty} n^{-1} log_q m_i$. More generally, we may define

Definition : Let **limit sup of the rate of information** be

$$\alpha(\delta) = lim\ sup\ n^{-1} log_q A(n, \delta n)$$

■

We shall define the **entrop function** $H_q(x)$ as

Definition : For $0 \leq x \leq \frac{q-1}{q}$, we define the *entropy function* $H_q(x)$ as

$$\begin{aligned} &H_q(0) = 0 \\ &H_q(x) = xlog_q(q-1) - xlog_q x - (1-x)log_q(1-x), \qquad x \neq 0 \end{aligned}$$

■

We need the following well-known Stirling's formula:

$$log\ n! = nlog\ n - n + O(log\ n)$$

as n$\mapsto \infty$. We have the following propositions.

Lemma 1.8.2 *Let* $0 \leq \lambda \leq \frac{q-1}{q}$ *and* $q \geq 2$. *Then*

$$lim_{n \mapsto \infty}\ n^{-1}\ log_q\ \mathbf{V}_q(n, \lfloor \lambda n \rfloor) = H_q(\lambda)$$

Proof. Let r=$\lfloor \lambda n \rfloor$. We separate the proof to two cases: Case 1. q=2. Case 2. q$\geq$ 3.

Case 1. We suppose that q=2. Then we have $\lambda \leq \frac{1}{2}$ and

$$\begin{aligned} &H_2(0) = 0 \\ &H_2(x) = -xlog_2 x - (1-x)log_2(1-x), \qquad x \neq 0 \end{aligned}$$

and

$$\mathbf{V}_2(n,r) = \sum_{i=0}^{r} \mathbf{C}_i^n$$

Using Stirling's formula, we deduce that

$$\begin{aligned} n^{-1}log\ (\sum_{i=0}^{r} \mathbf{C}_i^n) &\geq n^{-1}log\ \mathbf{C}_r^n \\ &= n^{-1}(nlog\ n - rlog\ r - (n-r)log\ (n-r) + O(n)) \\ &= log\ n - \lambda log(\lambda n) - (1-\lambda)log\ ((1-\lambda)n) + O(1) \\ &= H(\lambda) + O(1) \end{aligned}$$

So we prove one direction. For the other direction, we have

$$\begin{aligned} 2^{-nH(\lambda)} &= (\lambda)^{n\lambda}(1-\lambda)^{(1-\lambda)n} \\ &= (1-\lambda)^n(\frac{\lambda}{1-\lambda})^{\lambda n} \end{aligned}$$

Therefore we have

$$\begin{aligned} &2^{-nH(\lambda)} \sum_{0\leq i\leq \lambda n} \mathbf{C}_i^n \\ &= \sum_{0\leq i\leq \lambda n} \mathbf{C}_i^n (1-\lambda)^n (\frac{\lambda}{1-\lambda})^{\lambda n} \\ &\leq \sum_{0\leq i\leq \lambda n} \mathbf{C}_i^n (1-\lambda)^n (\frac{\lambda}{1-\lambda})^{i} \\ &= \sum_{0\leq i\leq \lambda n} \mathbf{C}_i^n \lambda^i (1-\lambda)^{n-i} \\ &\leq (\lambda + (1-\lambda))^n = 1 \end{aligned}$$

Or

$$\sum_{0\leq i\leq \lambda n} \mathbf{C}_i^n \leq 2^{nH(\lambda)}$$

After we take log on the above inequality, our case 1 follows.

Case 2. We suppose that q$\geq$ 3 or (q-1)$\geq$ 2. Then for $i \leq r$ we always have

$$\mathbf{C}_{i-1}^n (q-1)^{i-1} < \mathbf{C}_i^n (q-1)^i$$

For instance, for a proof, we have

$$\begin{aligned}
&\mathbf{C}^n_{i-1}(q-1)^{i-1} < \mathbf{C}^n_i(q-1)^i \Leftrightarrow i < (q-1)(n-i+1) \\
&\Leftrightarrow qi < (q-1)(n+1) \Leftrightarrow i < (n+1)\frac{q-1}{q} \\
&\Leftarrow i < n\frac{q-1}{q} \Leftarrow i \le r
\end{aligned}$$

It follows from Proposition 1.8.1 and the above that

$$\mathbf{C}^n_r(q-1)^r \le \mathbf{V}_q(n,r) \le (1+r)\mathbf{C}^n_r(q-1)^r$$

Now we simply imitate the proof of Case 1 to deduce the Case 2 (see exercise). ■

Lemma 1.8.3 *: Let n, d be positive integers and $d < n$. Then we have*

$$A(n,d) \ge q^n / V_q(n, d-1)$$

Proof. Let $\mathbf{C}$ be a code which gives $A(n,d)$. Then we know that there is no word in F_q^n with distance d or more to all words in $\mathbf{C}$. Therefore we have the following,

$$A(n,d) \times V_q(n, d-1) \ge q^n$$

■

Proposition 1.8.4 *(Gilbert-Varshamov's bound): We have*

$$\alpha(\delta) \ge 1 - H_q(\delta)$$

Proof. We have

$$\alpha(\delta) = lim\ sup\ n^{-1} log_q A(n, \delta n) \ge lim_{n\to\infty}(1 - n^{-1} log_q\ V_q(n, \lfloor \delta n \rfloor)) \qquad = 1 - H_q(\delta).$$

■

We may compare the above proposition to Shannon's Theorem. Consider the binary case, i.e., $q = 2$, then an $(n, m, n\delta)$ code with minimal Hamming distance $n\delta$ may correct $\frac{n\delta - 1}{2}$ errors (cf **Proposition 1.6.2**). In this case we observe that the rate of correcting error is about $\frac{\delta}{2}$ for large values of n. To compensate for a noisy channel of having a **rate of errors** $\wp$, we should choose a code with $\delta = 2\wp$. Then we have the following equations, note that $log_2(2-1) = 0$,

$$\alpha(2\wp) \ge 1 + 2\wp log_2\ 2\wp + (1 - 2\wp) log_2\ (1 - 2\wp) \qquad (Gilbert - Varshamov's bound)$$

For comparisson, we have Shannon's theorem,

$$R(M) \ge 1 + \wp\ log_2\ \wp + (1 - \wp) log_2\ (1 - \wp) - \epsilon \qquad (Shannon's\ Theorem)$$

Using elementary calculus, we may deduce the righthand side number (drop the small ϵ) in the second equation using Shannon's theorem is always bigger than the corresponding number in the first equaton as long as $2\wp < \frac{1}{2}$ which is valid for interested symmetric channels (see exercise). Therefore Shannon's theorem is stronger in this case. However, Shannon's theorem is an existence theorem, and (Gilbert-Varshamov's bound) is constructive. Therefore each has its own advantage. For more than 20 years, the Gilbert-Varshamov's bound which is a lower bound serves as a standard to measure all codes, it has only be surpassed by the *geometric Goppa codes* in the 90's (see Part IV).

Exercises

(1): Let C_n be the Hamming codes $[2^n - 1, 2^n - n - 1, 3]$. Let R_n be their rates of informations. Find $lim_{n \mapsto \infty} R_n$.

(2): Let C_n be the triple repetition code of length $3n$, i.e., a message $[a_1 a_2 \cdots a_n]$ is sent to $[a_1 a_1 a_1 a_2 a_2 a_2 \cdots a_n a_n a_n]$. Let R_n be their rates of informations. Find $lim_{n \mapsto \infty} R_n$.

(3): Find the maximal value of $H_q(x) - x$.

(4) Finish the proof of the case 2 in Lemma 1.8.2.

(5): Prove the last paragraph of the last section.

PART II

POLYNOMIAL, POWERSERIES ALGEBRA AND POLYNOMIAL CODES

Chapter 2

Finite Fields

It follows from *Shannon's Thereom* that we have to go to long codes to make better codes. However, a long code will introduce complexity for decoding. To overcome this complexity, we have to assume more algebraic structures. Furthermore, the coding theory that we have developed so far will only allow us to correct one error. To correct more errors, we have to study the polynomial ring $K[x]$ over a finite field K and the related topics. In this chapter, we will cover the essential material from finite field theory and the related topics for future use, beginning with an example to illustrate the usage of modern algebra in the next level of coding theory.

2.1 An Example 3

Before we study the abstract theory of finite fields, let us consider an example. Recall the $[7,4,3]$ Hamming code $\mathbf{C}$ having check matrix H as follows

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

Let us permute the rows to redefine H as follows:

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

Now consider the ring $\mathbf{F}_2[x]/(1+x+x^3)$=$\mathbf{R}$. It is easy to see that the polynomial $g(x) = 1+x+x^3$ is irreducible (for instance, g(0)≠0, and g(1)≠0 then $g(x)$ can not be factored), so then $\mathbf{R}$ is a field and isomorphic to $\mathbf{F}_2^3$ as vector spaces. If β is the image of x under the quotient mapping, then as a vector space we have

$$\mathbf{R} = \{a_0\beta^0 + a_1\beta^1 + a_2\beta^2 : a_i \in \ \mathbf{F}_2\}$$

We can drop the powers of β and write the vector space $\mathbf{R}$ as

$$\mathbf{R} = \{[000], [100], [010], [001],$$

$[110], [011], [111], [101]\}.whileasafield\mathbf{R}$=$\{0,\ 1,\ \beta, \beta^2, \beta^3, \beta^4, \beta^5, \beta^6\}$. The interesting thing about these two representations is that the ordering of elements are identical. for instance, the sixth element in the field representation is β^4 while

$$x^4 \equiv x + x^2 \ \ mod\ (1 + x + x^3)$$

Therefore

$$\beta^4 = 0 + 1\beta + 1\beta^2$$

and β^4 corresponds to [011] which is the sixth element in the vector space representation of $\mathbf{R}$. We may write $H = (1\beta\beta^2\beta^3\beta^4\beta^5\beta^6)^T$, and define Hamming code space as all words $a = [a_1a_2a_3a_4a_5a_6a_7]$ such that $a \times H = 0$, i.e., $\sum_1^7 a_i\beta^{i-1} = 0$. Alternatively we may write each word as a polynomial $a(x) = \sum a_i(x^{i-1})$ such that $deg\ a(x) \leq 6$ and $a(\beta) = 0 \in \mathbf{F}_8$. At this point, we may involve some known techniques of mathematics to handle the coding problem.

The condition $a(\beta) = 0 \in \mathbf{F}_8$ implies $a(x) = (1 + x + x^3) \times c(x)$. Therefore $deg\ c(x) \leq 3$ and $c(x) = c_1 + c_2x + c_3x^2 + c_4x^3$. The relation of $a(x)$ and $c(x)$ is as follows,

$$\begin{aligned} a(x) &= a_1 + a_2x + a_3x^2 + a_4x^3 + a_5x^4 + a_6x^5 + a_7x^6 \\ &= (1 + x + x^3)c(x) \\ &= (1 + x + x^3)(c_1 + c_2x + c_3x^2 + c_4x^3) \\ &= c_1 + (c_1 + c_2)x + (c_2 + c_3)x^2 + (c_1 + c_3 + c_4)x^3 \\ &\quad +(c_2 + c_4)x^4 + c_3x^5 + c_4x^6 \end{aligned}$$

We have the following *parametric form* of the code space with c_i's as the parameters,

$$\begin{aligned} &a_1 = c_1,\ a_2 = c_1 + c_2 \\ &a_3 = c_2 + c_3,\ a_4 = c_1 + c_3 + c_4 \\ &a_5 = c_2 + c_4,\ a_6 = c_3 \\ &a_7 = c_4 \end{aligned} \tag{3}$$

After we eliminate the parameters c_i's, we have the following system of defining equations

$$\begin{aligned} a_1 &= a_4 + a_6 + a_7 \\ a_2 &= a_4 + a_5 + a_6 \\ a_3 &= a_5 + a_6 + a_7 \end{aligned} \tag{4}$$

Therefore we have $[a_4a_5a_6a_7] \times G = [a_4a_5a_6a_7a_1a_2a_3]$ where G is the following matrix,

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

Note that G is the same generator matrix used in the introduction.

We can stay within the ring of polynomial $\mathbf{F}_2[x]$ by starting with message $[c_1 \cdots c_4]$, forming the polynomial $c(x) = \sum_1^4 c_i x^{i-1}$ with

$$a(x) = (1 + x + x^3)c(x) = \sum_1^7 a_i x^{i-1}$$

and to send the encoded message as the coefficients $[a_1 \cdots a_7]$. The receiver will check the received $[a'_1 \cdots a'_7]$ and compute $\sum_1^7 a'_i \beta^{i-1} = 0$ for $x = \beta^j$ for some j. Assume that there is at most one error in the received word. If the result of the preceding calculation is 0, then there is no error. Otherwise, there is an error at the $(j+1)$th spot of $[a'_1 \cdots a'_7]$. After correct the error and recover the original polynomial $a(x)$, one has the polynomial $c(x) = a(x)/(1+x+x^3)$ and the original message $[c_1 \cdots c_4]$. We will call the polynomial $g(x) = 1 + x + x^3$ the *generator polynomial*, and the polynomial $h(x) = (1+x^7)/g(x) = 1+x+x^2+x^4$ the *check polynomial.* Note that $a(x)$ is a code word iff $a(\beta) = 0$ iff $a(x) = c(x)g(x)$ iff $(1 + x^7)|a(x)h(x)$.

The above example illustrates that Hamming code can be discussed purely in terms of polynomials. This is the next level of development in coding theory. Before we continue our discussion of $K[x]$, we must have some understanding of the finite field K.

2.2 The Finite Field $\mathbf{F}_q$

We wish to discuss some basic structure of a finite field F. One of the basic property we will assume is the following theorem in *Theory of Groups*, vol 1 pg 147, [10] by A.Kurosh,

Theorem 2.2.1 *(Fundamental Theorem of Finitely Generated Abelian Group): Given any finitely generated abelian group G. Then G is isomerphic to $\prod Z \prod(Z_{c_i})$ as abelian additive group, where the $\prod$ has only finitely many arguments and $c_i > 0$ for all i.*

Proof. See the reference. ∎

We have the following easy propositions for abelian groups,

Proposition 2.2.2 *: (1). Let $G = Z_a \oplus Z_b$ and $d = lcm(a, b)$. Then every element $x \in G$ satisfies*

$$dx = 0$$

(2). Let $G = Z_a \oplus Z_b$ where a, b are coprime. Then G is cyclic of order ab.

Proof. (1) It is easy.

(2). Let c, d be selected such that $ac + bd = 1$. Let x be a generator of Z_a anf y be a generator of Z_b. We claim that $z = (dx, cy)$ is a generator $Z_a \oplus Z_b$.

We have the folloowing computations,

$$az = (adx, acy) = (0, acy) = (0, y(1 - bd)) = (0, y + (-bd)y) = (0, y)$$

similarly we have $bz = (x, 0)$. Therefore z generates G and G is cyclic. It is easy to show that $order(G) = ab$.

■

Proposition 2.2.3 : *Let G be a finitely generated multiplicative group of a finite field $\mathbf{K}$. Then G is cyclic.*

Proof. Since $\mathbf{K}$ is abelian with respect to multiplication, G is a finitely generated abelian group which is finite. We have that G is isomerphic to $\prod(Z_{c_i})$. Note that this ismorphism will change the multiplication of G to the summation in $\prod(Z_{c_i})$. Let us make an induction on the number factors n. In general, let us consider any finite group G which can be written as $\prod_{i=1}^{n}(Z_{c_i})$. If $n = 1$, then our proposition is clearly true. Let us assume that the proposition is true for any number $m < n$ of factors. If there at least two a_i, a_j with $d = lcm(a_i, a_j) < a_i a_j$, then all elements in the subgroup $(Z_{a_i} \oplus Z_{a_j})$ which has $a_i a_j$ elements satisfies

$$x^d = 1,$$

then the above polynomial will have more tham d distinct elements as solutions which is impossible for a field. Now it follows from a preceding proposition that $Z_{c_i} \oplus Z_{c_j} = Z_{c_i c_j}$. After this recombination, the factorization has only $n-1$ factors, and our proposition follows.

■

We have the following proposition which gives us lot of information about the structures of finite fields,

Proposition 2.2.4 : *Let K be a finite field of p^n elements in an algebraic closure Ω of Z_p. Then $K^* = K \setminus 0$ is a multiplicative cyclic group of order $p^n - 1$, and K is the solution set in Ω of the following equation,*

$$x(x^{p^n - 1} - 1) = x^{p^n} - x = 0$$

Furthermore all solutions of the following equation equation in Ω are distinct and the collection of them is a field,

$$x^{p^n} - x = 0.$$

If L is another finite field in Ω with p^m elements with $n|m$, then $L \supset K$ and

$$x^{p^n - 1} - 1 | x^{p^m - 1} - 1$$

Proof. The first statement follows trivially from the preceding proposition. For the second statement, let K_1 = the collection of all solutions of the equation $f(x) = x^{p^n} - x = 0$ in Ω. Since Ω is an algebraic closure of Z_p, then the equation $f(x) = 0$ splits completely. By considering the derivative of $f'(x) = 1$, the equation has no multiple root. Therefore K_1 consists of p^n elements. Moreover, let $y, z \in K_1$, then we have

$$(y+z)^{p^n} = y^{p^n} + z^{p^n} = y + z, \text{ and } (yz)^{p^n} = y^{p^n} z^{p^n} = yz$$

It is easy to check all requirements of a field to establish that K_1 is a field of p^n elements and therefore equal to K.

It is not hard to see the following proposition,

Proposition 2.2.5 *: We have $\Omega = \cup_{m>0} F_{p^m}$.*

Proof. . Let $\beta \in \Omega$. Then $\mathbf{Z}_p[\beta]$ is a finite extension of $\mathbf{Z}_p$. Therefore $\mathbf{Z}_p[\beta]$=$\mathbf{F}_q$ for some q. ■

Let us have the following common definition,

Definition : Let a mapping ρ be defined as $\rho(\alpha) = \alpha^p$. Then the mapping ρ is called the *Frobenius map.* ■

We have the following usual Proposition,

Proposition 2.2.6 *: The Frobenius map ρ is an automorphism of $\mathbf{F}_q$ of order m where $q = p^m$.*

Proof. . It is routine checking to see that ρ is an automorphism. Furthermore all elements $\beta \in \mathbf{F}_q$ satisfy

$$\beta^{p^m} = \beta$$

and $\rho^m(\beta) = \beta^{p^m} = \beta$. Therefore $\rho^m = id$. On the other hand if $\rho^\ell = id$ for some $\ell < m$ then we have $\beta^{p^\ell} = \beta$ for all $\beta \in \mathbf{F}_q$ which is impossible. ■

Corollary 2.2.7 *: Let $m = \ell s$. Then ρ^ℓ is an isomorphism of $\mathbf{F}_{p^m}$ which fixes $\mathbf{F}_{p^\ell}$. Note that the map ρ^ℓ as an isomorphism of $\mathbf{F}_{p^m}$ is of order s. Furthermore, it follows from Galois theory that any automorphism of $\mathbf{F}_{p^m}$ which fixes $\mathbf{F}_{p^\ell}$ is of the form $(\rho^\ell)^j$ for some suitable j.*

■

We will *assume Galois theory* and the second part of the above corollary. We define

Definition A field $\mathbf{K}$ is said to be *perfect* if either its characteristic is 0 or every element in $\mathbf{K}$ has a pth root in $\mathbf{K}$ where p is its characteristic. ■

Proposition 2.2.8 *A finite field is perfect.*

Proof. It is easy to see that the Frobenius map ρ is one-one. By the pigeon hole principle, it is onto. Therefore, for any given $\alpha \in \mathbf{K}$, there is an $\beta \in \mathbf{K}$ such that

$$\beta^p = \alpha$$

Clearly β is the pth root of α. ■

2.3 The computer program for finite fields

Let us assume that $p = 2$ in this section. Usually we think the set V_m of all $m-$bits as a vector space of dimension m over F_2. We want to put more algebraic structure on V_m. In fact, V_m can be considered as a finite field F_{2^m} over F_2. The multiplication is simple mathematically, while very messy in practise. We shall realize the multiplication in three ways as follows.

(1) **The table of logrithm** We shall only consider examples. The reader may generalize the following construction process to the general setup for any prime field Z/pZ with $p > 2$ and any finite field Z_{p^m}. Let us consider F_{2^4} over F_2. Let us write $F_{2^4} = F_2[\alpha]$ where α satisfies the equation $x^4+x+1 = 0$. We have the following list of powers of α, $\{1 = \alpha^0, \alpha, \alpha^2, \alpha^3, \alpha+1 = \alpha^4, \alpha^2 + \alpha = \alpha^5, \alpha^3 + \alpha^2 = \alpha^6$, $\alpha^3 + \alpha + 1 = \alpha^7, \alpha^2 + 1 = \alpha^8, \alpha^3 + \alpha = \alpha^9, \alpha^2 + \alpha + 1 = \alpha^{10}$, $\alpha^3 + \alpha^2 + \alpha = \alpha^{11}, \alpha^3 + \alpha^2 + \alpha + 1 = \alpha^{12}, \alpha^3 + \alpha^2 + 1 = \alpha^{13}$, $\alpha^3 + 1 = \alpha^{14}, 1 = \alpha^{15}\}$.

It is clear that $1, \alpha, \alpha^2, \alpha^3$ are linearly independent over F_2. Let us write them as [1000], [0100], [0010], [0001], then the above list of the powers of α may be re-written as [1000], [0100], [0010], [0001], [1100], [0110], [0011], [1101], [1010], [0101], [1110], [0111], [1111], [1011], [1001], [1000
One way to treat the problem of multiplication is for any two elements $[a_1a_2a_3a_4]$, $[b_1b_2b_3b_4]$, let us look up the list and find i, j such that

$$[a_1a_2a_3a_4] = \alpha^i$$
$$[b_1b_2b_3b_4] = \alpha^j$$

and we define

$$log_\alpha([a_1a_2a_3a_4]) = i$$
$$log_\alpha([b_1b_2b_3b_4]) = j$$

Then we have

$$log_\alpha([a_1a_2a_3a_4][b_1b_2b_3b_4]) = i + j$$

Let us find the residue value k of $i + j$ module $15 = 2^4 - 1$, i.e., $k = i + j \ mod \ 15$ and $0 \leq k < 15$. Let Now we may look up the above list of powers again to find the corresponding value of k determine the value of $[a_1a_2a_3a_4][b_1b_2b_3b_4]$. This method of looking up **the table of logrithm** of length $2^4 - 1$ (in general, for the field F_{2^m} a length $2^m - 1$, if m is large, say $m > 32$, then the table will be beyond the memory of an ordinary computer, and not be feasible).

(2) **The table of multiplication** Let us consider the impliment of a finite field K of p^m elements. We may consider the pair $K \supset Z_p$. We shall impliment the summation and the multiplication for the prime field Z_p first. The summation shall be the usual sum for integers and then mod p. For multiplication, there are two ways, either we multiply two numbers and then mod p, or we use a multiplication table, if the number of the elements in Z_p is not too big, and then use look up the table to achieve the multiplication. Note that the size of the table is $(1/2)(p-1) \times (p-2)$, after we delete all multiplications of the form $0 \times a = a \times 0 = 0$, and $1 \times a = a \times 1 = a$, and the size is the determining factor for the decision.

Let us program the field K which is uniquely determined by the number p^m of elements in K. i.e., let K be the collections of all vectors of the form $[a_1, a_2, \cdots, a_m]$ where $a_j \in Z_p$. we define the summation as the summation of vectors. For multiplication, consider the polynomial $x^{p^m} - x$. There are many polynomials factor $f(x) \in Z_p[x]$ which are irrecducible over Z_p and of degree m (see the exercise). Let us select such a polynomial $f(x)$. Let $v = [a_0, a_1, \cdots, a_{m-1}]$, and $u = [b_0, b_1, \cdots, b_{m-1}]$ be two elemen in K, then we write

$$f_v = a_0 + a_1x + a_2x^2 + \cdots + a_{m-1}x^{m-1}, \qquad f_u = b_0 + b_1x + b_2x^2 + \cdots + b_{m-1}x^{m-1}$$

and compute

$$f_v \times f_u \ mod(f(x)) = f_{vu}(x) = \sum_{j=0}^{m-1} c_jx^j$$

and we *define*: $v \times u = [c_0, c_1, \cdots, c_{m-1}]$. We have two ways to define the multiplication for the field K, if $(1/2)(p^m - 1)(p^m - 2)$ is small enough, then we may form a table of multiplication. Or if $(1/2)(p^n - 1)(p^n - 2)$ is too big, then for every multiplication, we shall go through the above equation of module polynomial $f(x)$ to find the product (see (3)).

Let us discuss the following example.

Example 4: Let K be a finite field of 2^3 elements. The $K \supset Z_2$ and is a vecotr space of dimension 3 over Z_2. Any element in K can be represented as $[a_0, a_1, a_2]$ where $a_j = 0, 1$. Let consider the polynomial $f(x) = 1+x+x^3$. It has no root in Z_2, therefore it is irreducible over Z_2 (see the exercise). Henceforth we may use it to define the multiplication in K. For the sake of notations, let use the ordinary numerial $\{0, 1, 2, \cdots, 7\}$ with each written as binary expansion, and use 0's to fill each to the length 3, as follows,

$$0 = [0, 0, 0],\ 1 = [1, 0, 0],\ 2 = [0, 1, 0],\ 3 = [1, 1, 0]$$

$$4 = [0, 0, 1],\ 5 = [1, 0, 1],\ 6 = [0, 1, 1],\ 7 = [1, 1, 1]$$

Then we have $2 \times 2 (= x \cdot x = x^2) = 4$. Similarly we may compute the following table,

$\times$	2	3	4	5	6	7
2	4	6	3	1	7	5
3	6	5	7	4	1	2
4	3	7	6	2	5	1
5	1	4	2	7	3	6
6	7	1	5	3	2	4
7	5	2	1	6	4	3

We use the above method to construct a multiplication matrix $(a_i a_j)$ for all pairs of nonzero elements (a_i, a_j) in the field. Once we have the table, then we have to do one table lookup to find out the result of multiplication of the pair (a_i, a_j).

Note that the table is symmetric, and the multiplication depends on the polynomial $f(x) = x^3 + x + 1$, for instance, $5 \times 3 = 4$ which comes from the following equation,

$$(x^2 + 1)(x + 1) = x^3 + x^2 + x + 1 = x^3 + x + 1 + x^2 = x^2 \ mod(x^3 + x^2 + 1)$$

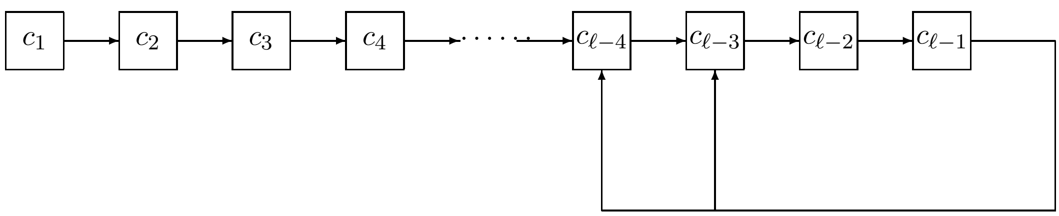
Note that $x^{8-1} - 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1) mod\ 2$. The other possible selection of the irreducible polynomial of degree 3 is $x^3 + x^2 + 1$. If we use it to define the multiplication, then we have $5 \times 3 = 2$. It is easy to see that the definition of multiplication depends on the selection of the irreducible polynomial of degree 3.

(3)**Linear Feedback Shift Register**

Let us consider another method. Let us assume that **q=2** in this subsection. iLet us consider F_{2^4}. Let the defining equation be,

$$f(x) = x^4 + x + 1 = 0$$

Note that we have $1 = -1$. Let $F_{2^4} = F_2[\alpha]$ and α satisfies the above equation. Let us represent $[a_1 a_2 a_3 a_4]$ as $a_1 + a_2\alpha + a_3\alpha^2 + a_4\alpha^3$ and $[b_1 b_2 b_3 b_4]$ as $b_1 + b_2\alpha + b_3\alpha^2 + b_4\alpha^3$. Then certainly, $[a_1 a_2 a_3 a_4][b_1 b_2 b_3 b_4]$ can be represented as $(a_1 + a_2\alpha + a_3\alpha^2 + a_4\alpha^3)(b_1 + b_2\alpha + b_3\alpha^2 + b_4\alpha^3) = (a_1 b_1) + (a_1 b_2 + a_2 b_1)\alpha + \cdots + (a_4 b_4)\alpha^6 = c_1 + c_2\alpha + \cdots + c_7\alpha^6$. Now we want to re-write the last expression as a polynomial in α of degree at most 3 by module out the defining equation. In general, we may reduce any polynomial $\sum_{i+1}^{\ell} c_i \alpha^{i+1}$ to a polynomial of α of degree at most 3 using a *linear feedback shift register* which simulates the module out the defining equation as follows,



LFSR for $x^4 = x + 1$

The process is that assume $c_\ell = 1 \neq 0$, then $c_{\ell-4} \to c_{\ell-4} + 1$ and $c_{\ell-3} \to c_{\ell-3} + 1$. The rest terms will stay. The next step is pushing all terms one step rightward. The term $c_{\ell-1}$ will fall off the horizontal line. If it is 0, then we do nothing and we push again. If $c_{\ell-1}$ is 1, then we use the above diagram to feed back and make changes of the terms. We keep making shifts until there are four terms,c_1, c_2, c_3, c_4, left. What left is the result of multiplication.

Exercises

(1): Let $\mathbf{\Omega}$ be an algebraic closure of $\mathbf{F}_p$ where p is a prime number. Show that there is a unique field of order $p^m \subset \mathbf{\Omega}$.

(2): Find an quadratic irreducible equation $f(x)$ over $\mathbf{F}_p$.

(3): Show that a polynomial $f(x)$ of degree n is irreducible over $\mathbf{F}_p \Leftrightarrow$

$$(1).f(x) \mid x^{p^n}$$
$$(2).(f(x), x^{p^m}) = (1), \ \ for\ all\ m < n$$

(4): Let α be a root of $x^2 + x + 1 = 0$ and β be a root of $x^2 + x + \alpha = 0$ in an algebraic closure Ω of $\mathbf{F}_2$. Show that $\mathbf{F}_4 = \mathbf{F}_2[\alpha]$ and $\mathbf{F}_{16} = \mathbf{F}_4[\beta]$.

(5): We use the notations of the preceding problem. Show that every element $\gamma \in \mathbf{F}_{16}$ can be expressed as $\gamma = a_3\alpha\beta + a_2\beta + a_1\alpha + a_0$ where $a_i = 0, 1$ are elements in $\mathbf{F}_2$. We may represent γ by an integer $a_3 2^3 + a_2 2^2 + a_1 2 + a_0$. Find γ^2 if γ is represented by 13.

(6): Imitate section 2.1 to find a polynomial code form for *Hamming Code* $[15, 11, 3]$.

(7): Find all group generators for $\mathbf{F}_{\mathbf{2^4}}$.

(8): Over the finite field $\mathbf{F}_{p^n}$, show that

$$(\alpha + \beta)^p = \alpha^p + \beta^p$$

(9): Find an element α which generates the multiplicative group $\mathbf{F}_{p^2}$ and find its defining equation.

(10): Let $\delta \in \mathbf{F}_{2^4}$ be a non-zero element. Find δ^{-1}.

2.4 The Total Quotient Rings

In Mathematics, we have **ring theory** after the **Group theory**. By definition, a **ring** is a mathematical object which satifies the following definition,

Definition : A **ring** $(R, +, \times)$ is a set R with two binary operations $+$ and $\times$, such that R is closed under those two operations, and they satisfies the following,

(1) *Group law* : $(R, +)$ is an abelian group.

(2) *Associative law* : we always have $a \times (b \times c = (a \times b) \times c$.

(3) *Distributive law* : we always have $a \times (b + c) = a \times b + a \times c$. and $(a + b) \times c = a \times c + b \times c$.

Sometimes we use $\cdot$ in the place $\times$. If $a \cdot b = b \cdot a$ always, then we say it is a ring. If it is commutative, and there is a multiplicative identity e, then we say it is a commutative ring with identity. In this book, all rings are **commutative ring with identity**. ■

Let **R** be a ring i.e., a commutative ring with identity in this book. A non-zero element α in **R** which is not a zero-divisor i.e., $\beta\alpha = 0, \Rightarrow \beta = 0$, will be called a *regular element.* The set of all regular elements in R does not contain zero and is closed under multiplication. When we take the set **S** of all regular elements as the set of *denominators*, then the quotient ring= $\{\frac{r}{s} : r \in R, s \in S\}$ is called the *total quotient ring* $\mathbf{R}_S$.

Example 5: Let **Z** be the ring of integers. Then its total quotient ring is the field of rational numbers **Q**. ■

Example 6: Let **R** be the residue class ring **K**[x]/(g(x)). Then an element $\overline{f(x)}$ is a regular element $\Leftrightarrow$ g(x) and f(x) are coprime. In that case, there are elments h(x) and r(x) such that

$$h(x)g(x) + f(x)r(x) = 1.$$

Therefore

$$\frac{1}{\overline{f(x)}} = \overline{r(x)}$$

It is easy to see that the total quotient ring of **K**[x]/(g(x)) is itself. Especially, if $f(x) = \prod(x-\beta_i)$ with $g(\beta_i) \neq 0$ where $\{\beta_i\}$ are all distinct, then any element $\overline{n(x)}/\overline{f(x)} \in \mathbf{K}[x]/(g(x))$ can be written as deg n(x)< deg g(x) and

$$\overline{n(x)}/\overline{f(x)} = \sum \frac{c_i}{\overline{(x - \beta_i)}}$$

■

2.5 The Ring $\mathbf{F}[x]$

Let us use **R** to denote the ring $\mathbf{F}[x]$ for any field **F**. We have the following definition.

Definition : Let a polynomial f(x)=$\sum_i c_i x^i$ be given. A degree function, deg(f(x)), of any polynomial f(x) is as follows,

$$deg(f(x)) = \begin{cases} max\{i : c_i \neq 0\} & if\ f(x) \neq 0 \\ -\infty & if\ f(x) = 0 \end{cases}$$

■

We have the following basic properties of deg(f(x)).

$$deg(f(x)g(x)) = deg(f(x)) + deg(g(x))$$
$$deg(f(x) + g(x)) \leq max(deg(f(x)), deg(g(x)))$$

we have the following Proposition,

Proposition 2.5.1 *: Let $g(x), \alpha_r(x), \alpha_u(x), \omega_r(x), f_u(x)$ be polynomials. If maximal(deg($\alpha_u(x)\omega_r(x)$), deg($\alpha_r(x)f_u(x)$)) $<$ deg($g(x)$). Then $\alpha_u(x)\omega_r(x) \equiv \alpha_r(x)f_u(x)$ mod $(g(x)) \Rightarrow \alpha_u(x)\omega_r(x) = \alpha_r(x)f_u(x)$.*

Proof. Trivial.

■

The following proposition is classically known and the proof is left to the reader.

Proposition 2.5.2 *(Euclidean algorithm): Given any two polynomials $f_1(x)$, $f_2(x) \neq 0$. There are polynomials $\beta_1(x)$ and $f_3(x)$ with deg($f_3(x)$)<deg($f_2(x)$) (note that $f_3(x)$=0 $\Leftrightarrow$ deg($f_3(x)$)=$-\infty$) and deg($\beta_1(x)$) $\leq$ deg($f_1(x)$)$-$deg($f_2(x)$)$\}$ such that*

$$f_1(x) = \beta_1(x)f_2(x) + f_3(x)$$

■

The above proposition can be used to find the GCD of two polynomials f(x), g(x). The following process, known as the "long algorithm", is of interest for *decoding* purposes later in the book. The main point is that we use *the Euclidean algorithms* in the process, and *the Euclidean algorithm* is fast in computing.

Proposition 2.5.3 *(Long Algorithm): Let $f_1(x), f_2(x), f_3(x)$ be given as in the preceding proposition. If $f_3(x) \neq 0$, the following sequence of polynomials $f_i(x)$ for $i = 4, \cdots, s$ with all polynomials nonzero except possibly $f_s(x)$, can be defined inductively for $j = 4, \cdots, s-1$ as*

$$f_{j-1} = \beta_{j-1}(x)f_j(x) + f_{j+1}(x)$$

such that after we name n_j=deg($f_j(x)$), m_j=deg($\beta_j(x)$), we have (1) $n_{j+1} < n_j$. (2) $m_{j-1} = n_{j-1} - n_j$. Furthermore, after repeated back-substitute (see the proof), we have the following equation for $j \geq 3$,

$$\alpha_j(x)f_1(x) + \gamma_j(x)f_2(x) = f_j(x)$$

with deg($\alpha_j(x)$) $\leq n - n_{j-1}$ and deg($\gamma_j(x)$) $\leq n - n_{j-1}$ where n= max(deg($f_1(x)$), deg($f_2(x)$)).

Proof. We may apply the Euclidean algorithm to the pair $f_2(x), f_3(x)$ and so on, until we reach the case that $f_s(x) = 0$. The first part of the proposition is routine. Suppose that we have the second part for $j = 4, \cdots, \ell$ with $\ell < s$. Then we have the following three equations,

$$\begin{aligned} f_{\ell-1} &= \beta_{\ell-1}(x)f_\ell(x) + f_{\ell+1}(x) \\ \alpha_{\ell-1}(x)f_1(x) + \gamma_{\ell-1}(x)f_2(x) &= f_{\ell-1}(x) \\ \alpha_\ell(x)f_1(x) + \gamma_\ell(x)f_2(x) &= f_\ell(x) \end{aligned}$$

Substituting the last two equations to the first one and collecting coefficients, we get the following equations,

$$\begin{aligned} \alpha_{\ell+1}(x) &= \alpha_{\ell-1}(x) - \alpha_\ell(x)\beta_\ell(x) \\ \gamma_{\ell+1}(x) &= \gamma_{\ell-1}(x) - \gamma_\ell(x)\beta_\ell(x) \end{aligned}$$

Note that

$$
\begin{aligned}
&deg(\alpha_{\ell-1}(x)) \leq n - n_{\ell-2} \leq n - n_\ell \\
&deg(\alpha_\ell(x)\beta_{\ell-1}(x)) \leq n - n_{\ell-1} + n_{\ell-1} - n_\ell \\
&deg(\gamma_{\ell-1}(x)) \leq n - n_{\ell-2} \leq n - n_\ell \\
&deg(\gamma_\ell(x)\beta_\ell(x)) \leq n - n_{\ell-1} + n_{\ell-1} - n_\ell
\end{aligned}
$$

Therefore we have

$$
\begin{aligned}
&deg(\alpha_{\ell+1}(x)) \leq n - n_\ell \\
&deg(\gamma_{\ell+1}(x)) \leq n - n_\ell
\end{aligned}
$$

■

The earlest application of the above theorem is to the *number theory.* One of the archievements of Zu (AD 430-501) is the approximate value of $\pi = \frac{335}{113}$. A possible explanation is that the continuous fraction of π is the folloowing,

$$\pi = 3 + \cfrac{1}{7 + \frac{1}{15 + \frac{1}{1+}}}$$

We get the sequence of numbers $\{3, \frac{22}{7}, \frac{333}{106}, \frac{335}{113}, \cdots\}$. 3 is known to King Soloman, $\frac{22}{7}$ was known to Archimedes, and Zu's number is among them. From the number theory, we know that this sequance give us the best possible approximations to π. For instance, if we restrict the denominator to be less than 1000, than $\frac{335}{113}$ is the best approximation to π. When we apply the long algorithm, we have to know the place to stop.

Now Y.Sugiyama, M.Kasahara, S.Hirasawa and T.Namekawa [32] noticed the place to stop for the above *long algorithm* for a modern application of the second part of the above proposition for decoding purposes.

From the above proposition, we can deduce that **R** is a *principal ideal ring*, i.e, each ideal is generated by one element,

LFSR

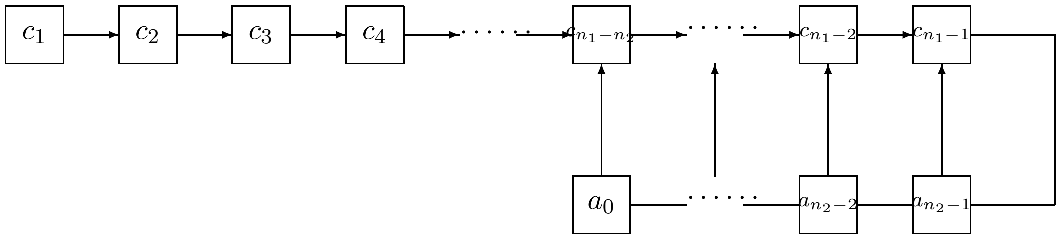
The *Euclidean algorithm* can implemented using LFSR (linear feedback shift register) in any computer effectively. Let us consider

$$f_1(x) = \beta_1(x)f_2(x) + f_3(x)$$

as in the statement of *Euclidean Algorithm.* Let

$$
\begin{aligned}
f_1(x) &= c_0 + c_1x + c_2x^2 + \cdots + c_{n_1-1}x^{n_1-1} + c_{n_1}x^{n_1} \\
f_2(x) &= a_0 + a_1x + \cdots + a_{n_2-1}x^{n_2-1} + x^{n_2}
\end{aligned}
$$

In general, polynomial divisions can be performed using LFSR. Let us consider a **simple case over $\mathbf{F}_2$**. We use the following LFSR to perform the division,



The process is that assume $c_\ell = 1 \neq 0$, then $c_{n_1-n_2} \to c_{n_1-n_2} + a_0$ and $c_{n_1-n_2+1} \to c_{n_1-n_2+1} + a_1$, and so on. The next step is pushing all terms one step rightward. The term c_{n_1-1} will fall off the horizontal line. If it is 0, then we do nothing and we push again. If c_{n_1-1} is 1, then we use the above diagram to feed back and make changes of the terms. We keep making shifts until there are n_2 terms, $c_0, c_2, \cdots, c_{n_2-1}$, left. What left is the result of the *Euclidean algorithm.* As an exercise, the reader is asked to set up an LFSR over F_3 using $f_2(x) = 1 - x + x^2 - x^3$.

Ideals

Proposition 2.5.4 *: Every ideal I of* $\mathbf{R}$ *is of the form (g(x)).*

Proof. We may use the preceding proposition to construct a generator g(x). For an existence proof, note that the zero ideal is generated by 0, otherwise, suppose I is non-zero. Let g(x) be a non-zero polynomial in I with the smallest degree. Then g(x) is a generator by Euclidean algorithm. ■

Corollary 2.5.5 *: The ring* $\mathbf{R}/I$ *is a principal ideal ring.*

Proof. Let $\bar{J}$ be an ideal in $\mathbf{R}$/I. Let J be the pre-image of $\bar{J}$ in $\mathbf{R}$. Then J is clearly an ideal. Let g(x) be its generator. Then clearly $\overline{g(x)}$ generates $\bar{J}$. ■

The following proposition is the Lagrange interpolation theorem.

Proposition 2.5.6 *: Let* $\{\beta_i\}_1^n$, $\{\alpha_i\}_1^n$ *be elements in* $\mathbf{F}$ *such that all* β_i*'s are distinct. Then there is an unique polynomial f(x) of degree at most* $n-1$ *such that* $f(\beta_i) = \alpha_i\ \forall\ i$*. Moreover f(x) is of the following form*

$$f(x) = \sum_i \alpha_i \frac{\prod_{j\neq i}(x-\beta_j)}{\prod_{j\neq i}(\beta_i - \beta_j)}$$

Proof. It is easy to see that the f(x) defined above satisfies the requirements of the proposition. Suppose g(x) is another. Then we have f(x)-g(x) will have n roots β_i and of degree at most $n-1$. Therefore f(x)-g(x)=0 or f(x)=g(x). ∎

Corollary 2.5.7 *: Let $\{\beta_i\}$ be a set of n distinct numbers. Let $\mathbf{P}_n$ be the set of all polynomials of degrees $\leq n-1$. The $\mathbf{P}_n$ is generated by $\{\prod_{j\neq i}(x-\beta_j)\}$.*

∎

We make the following simple generalization of the above Lagrange interpolation theorem.

Proposition 2.5.8 *: Let $\{\beta_i\}_1^m$, $\{\alpha_i\}_1^n$ be elements in $\mathbf{F}$ where $m \leq n$ and all β_i's are distinct and α_i's may not be all distinct. Furthermore, we have positive integers m_i (i.e., the multiplicities) such that $\sum_{i=1}^m m_i = n$. Then there is an unique polynomial f(x) of degree at most $n-1$ such that $f(x)-\alpha_i = (x-\beta_i)^{m_i} f_i(x) \forall\ i$ for some suitable polynomials $f_i(x)$.*

Moreover there are polynomials $g_i(x)$ with $deg(g_i(x)) \leq (m_i-1)$ satisfy

$$\sum_i g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j} = 1$$

such that f(x) is of the following form

$$f(x) = \sum_i \alpha_i g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j}$$

Proof. We will prove the equation

$$\sum_i g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j} = 1$$

first. We shall use the proof of the Chinese Remainder Theorem. Since the polynomials $\prod_{j\neq i}(x-\beta_j)^{m_j}$'s are co-prime, then they generate the unit ideal, i.e., we have

$$\sum_i h_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j} = 1$$

for some suitable $h_i(x)$. We may not take h_i to be g_i, because the degree restriction on g_i may not be satisfied. Let r=max$\{deg(h_i(x))-m_i\}$. If $r<0$, then we just let $g_i(x)=h_i(x)$. Note that the degree restriction is satisfied. If $r\geq 0$, let $s=$ the number of $h_i(x)$ such that $deg(h_i(x))-m_i=r$. We will make a double induction on r, s, i.e., we will reduce the number s to 0, then r will automatically drop. When r drops to negative, then we find $g_i(x)$'s.

Note that $s>1$. Otherwise, $s=1$, there is a unique term of highest degree, it can not be canceled by any other term, and the above equation can not be satisfied. Therefore $s\geq 2$. Let us pick up any two terms of the highest degree, say corresponding to $h_i(x)$ and $h_j(x)$. We have $r = deg\ h_i(x)-m_i = deg\ h_j(x)-m_j$. For any c, then $h_i(x), h_j(x)$ can be replaced by $h_i(x)+cx^r(x-\beta_i)^{m_i}$ and $h_j(x)-cx^r(x-\beta_j)^{m_j}$ respectively and the above equation is still satisfied. We may select c such that $\deg(h_i(x) + cx^r(x-\beta_i)^{m_i})$ is smaller. Thus that at least one terms drop out from the collection of the highest terms. we reduce the number s at least by 1. When the number s drops to zero, then r must drop. So we find $g_i(x)$'s by double induction.

Now we have

$$\sum_i g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j} = 1 \qquad \text{and}$$
$$\alpha_k(\sum_i g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j}) = \alpha_k$$

Therefore

$$\begin{aligned}
&(\sum_i \alpha_i g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j}) - \alpha_k\\
&= (\sum_i \alpha_i g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j}) - a_k((\sum_i g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j}))\\
&= (\sum_{i\neq k} \alpha_i g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j}) - a_k((\sum_{i\neq k} g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j}))\\
&= (x-\beta_k)^{m_k} f_k(x)
\end{aligned}$$

In other words, we may let $f(x) = \sum_i \alpha_i g_i(x)\prod_{j\neq i}(x-\beta_j)^{m_j}$. Furthermore, if there are another polynomial f*(x) with the same properties of f(x), namely

$$f(x) - \alpha_k = (x-\beta_k)^{m_k} f_k(x) \qquad \text{and} \qquad f^*(x) - \alpha_k = (x-\beta_k)^{m_k} f_k^*(x)$$

then f(x)−f*(x) will be divisible by $(x-\beta_i)^{m_i}$ for all i, i.e., f(x)− f*(x) will be divisible by $\prod_i (x-\beta)^{m_i}$ which has a degree n higher than $n-1$. We conclude f(x)−f*(x)=0. ∎

Exercises

(1): Prove the Chinese Remaider Theorem for $K[x]$, and use it to prove the Lagrange interpolation theorem.

(2): Find the total quotient ring of $\mathbf{Z}/4\mathbf{Z}$.

(3): Let $\chi_\gamma(x)$ be the characteristic function of $\gamma \in \mathbf{K}^n$ where $\mathbf{K}$ is a finite field, i.e., $\chi_\gamma(\gamma) = 1$ and $\chi_\gamma(a) = 0$ for all $a \neq \gamma$. Show that $\chi_\gamma(x)$ can be written as a polynomial in n variables.

(4): Let π be any $\mathbf{K}$ valued function on $\mathbf{K}^n$ where $\mathbf{K}$ is a finite field. Show that π can be written as a polynomial in n variables.

(5): Find two distinct polynomials $f(x), g(x) \in \mathbf{F}_p[x]$ such that $f(\gamma) = g(\gamma)$ for all $\gamma \in \mathbf{F}_p$.

2.6 Separability

One interesting criterion for a polynomial f(x) to have a multiple root in Ω is the derivative test. We define pure algebraically,

Definition : Let f(x)=$\sum_i a_i x^i$. Then the derivative $f'(x)$ is defined to be $f'(x) = \sum_i i a_i x^{i-1}$.

■

It is easy to see that the derivative obeys the following formal rules.

Proposition 2.6.1 : *The derivative operation has*

$$(1) : a' = 0$$
$$(2) : (f(x) + g(x))' = f'(x) + g'(x)$$
$$(3) : (f(x)g(x))' = f'(x)g(x) + f(x)g'(x)$$

Proof. Routine checking. ■

We have the following criterion.

Proposition 2.6.2 : *An element $\beta \in \Omega$ is a multiple root of f(x) $\Longleftrightarrow$ β is a root of f(x) and $f'(x)$.*

Proof. Let β be a multiple root of f(x). Then f(x)= $(x-\beta)^2$h(x) with h(x)$\in \Omega[x]$. Then we have

$$f'(x) = 2(x-\beta)h(x) + (x-\beta)^2 h'(x) = (x-\beta)g(x)$$

Therefore β is a root of f(x) and $f'(x)$. On the other hand, if β is a root of f(x) and $f'(x)$. Write f(x)=$(x-\beta)$r(x), we have

$$f'(x) = r(x) + (x-\beta)r'x(x)$$

We conclude that β must be a root of r(x), and f(x)=$(x-\beta)^2$s(x). ■

We have the following definition.

Definition A polynomial $f(x) \in \mathbf{K}[x]$ is said to be separable if it has no multiple root in an algebraic closure of **K**.

■

Proposition 2.6.3 *Over a finite field $\mathbf{F}_q$, every irreducible polynomial is separable.*

Proof. Let f(x) be an irreducible polynomial. If it has a multiple root, then the root must be a root of $f'(x)$. In other words, f(x) and $f'(x)$ must have a common factor. With the assumption of irreducibility of f(x), this can happen only if $f'(x)$ is the zero polynomial, i.e., f(x)=g(x^p) for some suitable g(x). Since a finite field is perfect, every coefficient of g(x) is a p-th power of some other element. It is easy to see that g(x^p)=h(x)^p for some suitable h(x). It contradicts with the irreducibility of f(x).

■

Let $q = p^\ell$. We wish to count the number of monic irreducible polynomials of degree r in $\mathbf{F}_{q^m}$[x]. Let the number be $\mathbf{I}_r$.

Proposition 2.6.4 *We have*

$$q^{ms} = \sum_{r|s} r\mathbf{I}_r$$

Proof. Let us fix an algebraic closure Ω of $\mathbf{F}_{q^m}$. Let $\alpha \in \mathbf{F}_q$ satisfying a monic polynomial $f_\alpha(x)$ of degree, say r. From finite field theory, we have $r \mid s$. From the preceding Proposition, we know that $f_\alpha(x)$ is separable. Any root of $f_\alpha(x)$ will generate a field extension over $\mathbf{F}_{q^m}$ of degree r. It follows from **Proposition 1.8.1** that this field extension is contained in $\mathbf{F}_{q^{ms}}$. We may group all roots (there are r of them) of $f_\alpha(x)$ together. Our Proposition follows easily. ■

Now we use the well-known Moebius inversion formula in number theory to find $\mathbf{I}_r$. For this purpose, we define

Definition The Moebius function μ is defined as

$$\mu(n) = \begin{cases} 1, & \text{if n=1} \\ (-1)^k, & \text{if n is a product of k distinct primes} \\ 0, & \text{otherwise} \end{cases}$$

■

The following theorem is well-known and its proof will be omitted.

Theorem 2.6.5 *(Moebius inversion formula):* $g(n)=\sum_{d|n} f(d) \Leftrightarrow f(n)=\sum_{d|n} \mu(d)g(\frac{n}{d})$.

Proof. (1). The Moebius function $\mu(n)$ satisfies

$$\sum_{d|n} \mu(d) = \begin{cases} 1, & \text{if n=1} \\ 0, & \text{otherwise} \end{cases}$$

(2).

$$\begin{aligned} \sum_{d|n} \mu(d)g(\frac{n}{d}) &= \sum_{d|n} \mu(\frac{n}{d})g(d) \\ &= \sum_{d|n} \mu(\frac{n}{d}) \sum_{k|d}(f(k)) = \sum_{k|n} \sum_{d|(n/k)} \mu_{\frac{n}{kd}} f(k) \\ &= \sum_{k|n} \sum_{d|(n/k)} \mu(d)f(k) = \sum_{k|n} \sum_{d|n/k)} \mu(d)f(k) \\ &= f(n) \end{aligned}$$

■

Recall that I_r is the number of monic irreducible polynomials of degree r in $F_{q^m}[x]$. It follows that we have

Proposition 2.6.6 *If $r = 1$, then $I_r > 0$. If r is a prime, then $r\mathbf{I}_r = (q^{mr} - q^m) > 0$. Otherwise, then $r\mathbf{I}_r = \sum_{d|r} \mu(d) q^{mr/d} > q^{mr}(1 - q^{-rm/2+1}) > 0$.*

Proof. It is clear that every linear polynomial is irreducible. We have the first part. It follows from the preceding theorem that

$$r\mathbf{I}_r = \sum_{d|r} \mu(d) q^{mr/d}$$

The second part follows at once. For the third part, we have $r \geq 3$ and

$$\begin{aligned} r\mathbf{I}_r &= \sum_{d|r} \mu(d) q^{mr/d} \\ &> (q^{mr} - \sum_{i \geq 2} q^{md/i}) \\ &> (q^{mr} - q^{mr/2+1}) \\ &= q^{rm}(1 - q^{-rm/2+1}) \\ &> 0 \end{aligned}$$

■

The above Proposition shows the existence of finite field F_{p^m} for any m. We have the following diagram, where lines indicate inclusive relation,

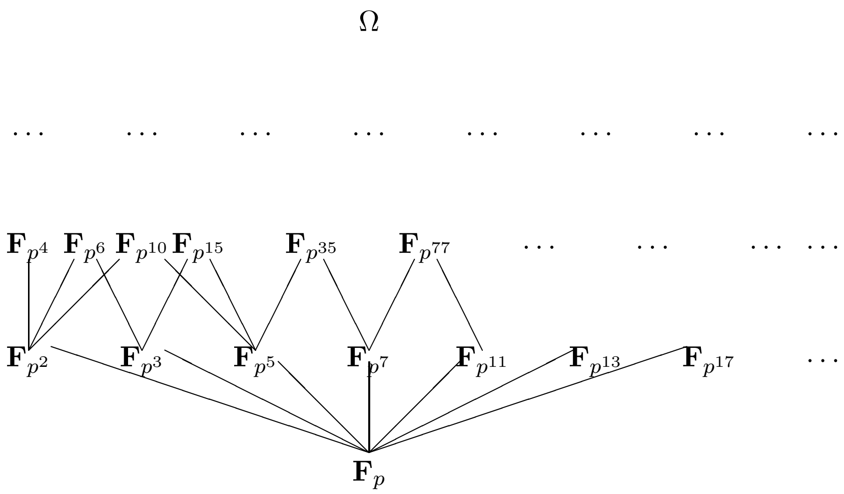


Figure 2

2.7 Power Series and Meromorphic functions

We shall consider formal power series in this section. Let us consider the expression $f(x) = \sum_{i=0}^{\infty} a_i x^i$ where a_i's are coefficients in the field $\mathbf{F}$. In the past, analysts considered the problem of evaluating the expression for $x = b$ and deducing many concepts of convergence, divergence etc. However, if we do not evaluate them (other than the trivial evaluation at $x = 0$), then we may simply treat them as algebraic items. Henceforth, the expressions of the form

$$f(x) = \sum_{i=0}^{\infty} a_i x^i$$

will be called *formal power series*, and the expressions of the form

$$f(x) = \sum_{i=-m}^{\infty} a_i x^i$$

will be called

$$`meromorphicfunctions.Allpowerserieswithcoefficientsin\mathbf{F}$$

$_q$ will be denoted by $\mathbf{F}$[[x]]. All meromorphic functions with coefficients in $\mathbf{F}$ will be denoted by $\mathbf{F}((x))$.

Let f(x)=$\sum_j a_j x^j$, g(x)=$\sum_k b_k x^k$. We define as usual,

$$f(x) \cdot g(x) = \sum_i (\sum_{j+k=i} a_j b_k) x^i$$
$$f(x) + g(x) = \sum_i (a_i + b_i) x^i$$

We have

Proposition 2.7.1 *: The sets* $\mathbf{F}[[x]] \subset \mathbf{F}((x))$ *are closed under the usual addition and multiplication.*

Proof. The proposition follows by routine checking. ■

Definition : Let f(x)= $\sum_i c_i x^i$ be a power series or a meromorphic function. We define the *order* of f(x), in symbol ord(f(x)), as

$$ord(f(x)) = \begin{cases} min\{i : c_i \neq 0\} & if\ f(x) \neq 0 \\ \infty & if\ f(x) = 0 \end{cases}$$

■

We have

Proposition 2.7.2 *: We have*

$$ord(f(x)g(x)) = ord(f(x)) + ord(g(x))$$
$$ord(f(x) + g(x)) \geq min(ord(f(x)), ord(g(x)))$$

Proof. Routine checking. ■

Proposition 2.7.3 : *We have f(x) is a unit in* $\mathbf{F}[[x]]$ *(i.e.,* $f(x)^{-1}$ *exists)* $\Longleftrightarrow f(0) \neq 0$.

Proof. ($\Longrightarrow$) If f(x) is a unit, then there exists g(x) such that f(x)g(x)=1. Then we have

$$ord(f(x)) + ord(g(x)) = 0$$

Therefore ord(f(x))=0 and $f(0) \neq 0$.

($\Longleftarrow$) If $f(0) = a \neq 0$, then $a^{-1}f(0) = 1$. We may write $a^{-1}f(x)$ as 1−g(x) with ord(g(x))≥ 1. We have

$$(a^{-1}f(x))^{-1} = 1 + \sum_{i=1}^{\infty} g(x)^i$$

It is easy to see that when we collect coefficients on the right hand side, we only have finite sums. Therefore the right hand side is a power series. It is then trivial to see that the inverse of f(x) exists. ■

A ring is said to be an integral domain if $f \cdot g = 0$, then either $f = 0$ or $g = 0$. We have the following,

Proposition 2.7.4 : *The set* $\mathbf{F}[[x]]$ *is an integral domain and* $\mathbf{F}((x))$ *is field.*

Proof. Routine checking. ■

Let f(x)=$\sum_{i=0}^{m} a_i x^i$, g(x)=$\sum_{i=0}^{n} b_i x^i$ be two polynomials of degrees m, n respectively. For simplicity let us assume that $b_0 = g(0) \neq 0$. Then $\frac{f(x)}{g(x)} \in \mathbf{F}_q$[[x]]. We have

$$\frac{f(x)}{g(x)} = \sum_{i=0} c_i x^i$$

The total number of coefficients of $\frac{f(x)}{g(x)}$ is (m+1+n+1). However, we may change f(x), g(x) by a common multiple of non-zero constant without changing $\frac{f(x)}{g(x)}$. Therefore, the number of independent coefficients should be (m+n+1). It is natural to ask if the first (m+n+1) coefficients $c_0, c_1, \cdots, c_{m+n}$ uniquely determine the rational function $\frac{f(x)}{g(x)}$. We have the following natural proposition which is significant for the computational purposes.

Proposition 2.7.5 : *Given any polynomial h(x) of degree m+n, there is at most one rational function* $\frac{f(x)}{g(x)}$ *where f(x), g(x) are of degrees at most* m, n *respectively with* $g(0) \neq 0$ *such that in the following equation in* $\mathbf{F}[[x]]$,

$$ord(\frac{f(x)}{g(x)} - h(x)) \geq m + n + 1$$

Furthermore (1) if such a rational function $\frac{f(x)}{g(x)}$ exists, then $\frac{f(x)}{g(x)} = \frac{f_1(x)}{g_1(x)}$ with $f_1(x), g_1(x)$ being of degrees at most $m-s, n-s$ respectively for some non-negative integer s, $\Longleftrightarrow$

$$ord(\frac{f_1(x)}{g_1(x)} - h(x)) \geq m+n-s+1$$

(2) if a factor $a(x)$ of degree u of $g(x)$ is known, then we have

$$ord(\frac{f(x)}{g(x)} - h(x)) \geq m+n-u+1$$

Proof. Let $\frac{f^*(x)}{g^*(x)}$ where f*(x), g*(x) are of degrees at most m, n respectively with $g^*(0) \neq 0$ and

$$ord(\frac{f^*(x)}{g^*(x)} - h(x)) \geq m+n+1$$

Then we have

$$ord(\frac{f(x)}{g(x)} - \frac{f^*(x)}{g^*(x)}) \geq m+n+1$$

On the other hand, we have

$$\frac{f(x)}{g(x)} - \frac{f^*(x)}{g^*(x)} = \frac{f(x)g^*(x) - f^*(x)g(x)}{g(x)g^*(x)}$$

Note that g(0)g*(0)$\neq$ 0. We conclude

$$ord(f(x)g^*(x) - f^*(x)g(x))) \geq m+n+1$$

Note that f(x)g*(x)$-$f*(x)g(x) is a polynomial of degree at most m+n. Hence it must be 0. We have

$$\frac{f(x)}{g(x)} = \frac{f^*(x)}{g^*(x)}$$

Thus we have the uniqueness. The proofs of last parts (1), (2) of the proposition are similar to the above and left to the reader.

■

Example 7: For a polynomial h(x) in the preceding proposition, there may not be a rational function $\frac{f(x)}{g(x)}$ with the said property. For instance, let h(x)=$1+x^4$, $m=n=2$, then there is no $\frac{f(x)}{g(x)}$ with $deg(f(x)) \leq 2$ and $deg(g(x)) \leq 2$ such that

$$ord(\frac{f(x)}{g(x)} - h(x)) \geq m+n+1 = 5$$

Suppose the contrary, multiplying by g(x) to the equation $\frac{f(x)}{g(x)} - h(x)$, it will produce

$$f(x) = g(x) + g(0)x^4 \; mod \; (x^5)$$

namely $f(x) = g(x)$ which is clearly impossible. ■

Exercises

(1): Factor $x^{16} - x$ over $\mathbf{F}_2$ into a product of irreducible polynomials.

(2): Given a finite field $\mathbf{F}_p$. Show that the following polynomial $f(x)$

$$f(x) = 1 + \cdots + \frac{x^i}{i!} + \cdots + \frac{x^{p-1}}{(p-1)!}$$

is separable.

(3): Let K be a field with $\mathrm{ch}(K)$ positive. Show that $x^p - x - a$ is either irreducible or splits completely into linear factors where $a \in K$.

(4): Let $z = x^2 + x^3$. Show that $[\mathbf{F}_2((x)) : \mathbf{F}_2((z))] = 2$ and find the equation satisfied by x over $\mathbf{F}_2\{\{z\}\}$.

(5): Let $\mathbf{K}$ be a field of characteristic p and x, y be symbols. Show that $\mathbf{K}(x, y)$ is inseparable over $\mathbf{K}(x^p, y^p)$.

Chapter 3

Polynomial Codes

As pointed by Shannon's theorem, a good code can only be a long code. While a long code may introduce complications for decoding, we have to use good algebraic structures to counter the complications in decoding.

From now on, we shall fix the ground field $\mathbf{F}_q$. A linear coding is essentially defined by a subspace $\mathbf{C}$ of a n-dimensional vector space $\mathbf{F}_q^n$. It is easy to see that $\mathbf{F}_q^n$ can be represented by $\mathbf{F}_q$[x]/(f(x)) for any polynomial f(x) of degree n. Furthermore in this representation we have the multiplicative structure of a ring. We shall use the concepts of coding theory in the context of the ring $\mathbf{F}_q$[x]/(f(x)). For instance, we have the following definition.

Definition : The Hamming weight of an element $\bar{h}(x) \in \mathbf{F}_q$[x]/(f(x)) is the Hamming weight of $[c_0c_1 \cdots c_{n-1}]$ where $\bar{h}(x) = \sum_{i=0}^{n-1} c_i x^i$ with $n = deg(f(x))$. ■

There is a convenient way to consider only **code subspaces** defined by ideals $\bar{I}$ in $\mathbf{F}_q$[x]/(f(x)), and the elements in $\bar{I}$ are called code words. Note that $\mathbf{F}_q$[x] is a principal ideal ring,

hence let $\bar{I}$ be any ideal in $\mathbf{F}_q$[x]/(f(x)) and I be its pull back in $\mathbf{F}_q$[x], i.e. $I = \{g(x) : \bar{g}(x) \in \mathbf{F}_q[x]/(f(x))\}$, then I is a principal ideal. We write $I = (h(x))$. It is clear that $\bar{I} = (\bar{h}(x))$. So we conclude that $\mathbf{F}_q$[x]/(f(x)) is a principal ideal ring. Furthermore, any $h(x)$ is co-prime to $f(x) \Leftrightarrow (\bar{h}(x)) = (1)$. Since an ideal $I = (f(x), h(x))$ is generated by the common greatest common divisor $h^*(x)$, we must have that $h^*(x)$ is a factor of $f(x)$. It is easy to see that any ideal $\bar{I} \neq (1)$ of $\mathbf{F}_q$[x]/(f(x)) is of the form $(\bar{h}(x))$, where $h(x)$ is a factor of $f(x)$ in $\mathbf{F}_q$[x].

If we take $f(x)$ to be $x^n - 1$, then we have a *cyclic* code in the following sense,

Definition : A code $\mathbf{C}$ of length n is said to be *cyclic* iff $[a_1a_2 \cdots a_n] \in \mathbf{C} \Leftrightarrow [a_na_1 \cdots a_{n-1}] \in \mathbf{C}$. ■

Example 8: Let us consider *Hamming code* $[7, 4, 3]$ as in section 2.1. It is a cyclic code and can be represented by the ideal $(x^3 + x + 1)$ in $\mathbf{F_2}[x]/(x^7 + 1)$. ■

A cyclic code is rich in algebraic structures, we have the following equivalent characterizations.

Proposition 3.0.6 *: Any ideal $\bar{I} \neq (1)$ in $\mathbf{F}_q[x]/(x^n-1)$ defines a cyclic code. Conversely any cyclic code of length n can be represented by an ideal $\bar{I}$ in $\mathbf{F}_q[x]/(x^n-1)$.*

Proof. . Let an ideal $\bar{I} \neq (1)$ in $\mathbf{F}_q[x]/(x^n-1)$ and a code word $c(x) = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1} \in \bar{I}$. Then $xc(x) \in \bar{I}$ and $xc(x) = c_{n-1} + c_0 x + \cdots + c_{n-2}x^{n-1}$.

Conversely. if $\mathbf{C}$ is cyclic of length n, let $c(x) = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}$ forms a set $\bar{I}$ for all $[c_0 c_1 \cdots c_{n-1}] \in \mathbf{C}$. The cyclic property of $\mathbf{C}$ implies $xc(x) \in \bar{I}$, hence $x^\ell c(x) \in \bar{I}$. It is easy to see that $\bar{I}$ is an ideal in $\mathbf{F}_q[x]/(x^n-1)$ and $\mathbf{C}$ can be represented by $\bar{I}$. ∎

The case $p \mid n$ in the above is not interesting, because then we have $x^n - 1 = (x^{n/p}-1)^p$, therefore $(x^{n/p}-1)$ is a non-zero code word, while p-th power of it is zero, this is against intuition, which is a *degenerate case*. From now on we will assume that n, p are coprime. Note that every ideal $\bar{I}$ can be generated by a factor of $x^n - 1$.

Definition : A monic polynomial g(x) which is a factor of $x^n - 1$ is called the **generator polynomial** of the cyclic code **C**=$(\bar{g}(x))$ and the polynomial h(x)=$(x^n-1)/g(x)$ is called **check polynomial** of **C**. ∎

Let Ω be an algebraic closure of $\mathbf{F}_q$. We will denote $f(x) = x^n - 1$. Under our assumption n, p are co-prime, we have

$$f'(x) = nx^{n-1}$$

which has only 0 as a root, furthermore 0 is not a root of $f(x)$. Therefore, the polynomial $x^n - 1$ will have no multiple root in Ω by the derivative test (cf **Proposition 2.6.2**). Therefore its divisor g(x), the *generator polynomial*, will have no multiple root. Let $\{\gamma_j\}$ be the set of all roots of g(x) in Ω. Then it is clear that $r(x) \in$ the ideal (g(x)) iff $r(\gamma_j) = 0$ for all j. Thus the ideal (or subspace **C**) can be defined by $\{r(x) : r(\gamma_j) = 0 \ mod \ (x^n-1), \forall j\}$. In fact, we may not have to take all roots of $g(x)$ in Ω, a partial set of roots may be enough.

Proposition 3.0.7 *: Given any set $\{\gamma_j\} \subset \Omega$ such that any γ_j is a root of $x^n - 1$ for a fixed n, let g(x) be the least common multiple of all monic irreducible polynomials satisfied by $\{\gamma_j\}$ for all j. Then g(x) is the generator polynomial of a cyclic code $\mathbf{C}$ in $\mathbf{F}_q[x]/(x^n-1)$.*

Proof. Let $g_i(x)$ be the monic irreducible polynomial satisfied by γ_i. Then it is clear that $g_i(x) \mid x^n - 1$ for all i. Therefore $x^n - 1$ is a common multiple of all $g_i(x)$ and g(x)$\mid x^n - 1$. ∎

3.1 BCH Codes

Let us recall the significant property of the Hamming code as described by **Proposition 1.6.1**, which concludes that the Hamming distance of such a code is 3 by showing any two column vectors are linearly independent while some 3 column vectors are linearly dependent. There are well-known matrices called the Verdermonde matrices whose column vectors are linearly independent. We have the following proposition.

Proposition 3.1.1 *: Let M be the following matrix,*

$$M = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & ... & 1 \\ x_1 & x_2 & x_3 & x_4 & x_5 & ... & x_n \\ x_1^2 & x_2^2 & x_3^2 & x_4^2 & x_5^2 & ... & x_n^2 \\ x_1^3 & x_2^3 & x_3^3 & x_4^3 & x_5^3 & ... & x_n^3 \\ ... & ... & ... & ... & ... & ... & ... \\ x_1^{n-1} & x_2^{n-1} & x_3^{n-1} & x_4^{n-1} & x_5^{n-1} & ... & x_n^{n-1} \end{pmatrix}$$

Then we have

$$det\ M = \prod_{i>j}(x_i - x_j)$$

Proof. We treat all x_i as symbols first. Subtracting the first column from the second column, $\cdots$, the n-th column , we may extract $x_i - x_1$ from the i-th column, we conclude that $\prod_{i>1}(x_i - x_1) \mid \det(M)$. By symmetry, we conclude

$$\prod_{i>j}(x_i - x_j) \mid det(M)$$

Since both sides are polynomials of degrees $n(n-1)/2$, they can only differ by a constant. Comparing the coefficients of $\prod_{i\geq 2} x_i^{i-1}$ on both sides, we conclude that they must be equal. Since the proposition is true for x_i as symbols, it must be true for x_i as any elements.

■

Let us further consider the ring $\mathbf{F}_q[x]/(x^n - 1)$ with the usual assumption that n, p are coprime. For the purpose of coding theory we have the following proposition.

Proposition 3.1.2 *: Let γ be a primitive n-th root of unity (i.e., γ is a root of $x^n - 1$ and not a root of $x^s - 1$ for any $0 < s < n$) in Ω. Let g(x) be the least common multiple of all monic irreducible polynomials of $\gamma^\ell, \gamma^{\ell+1}, \cdots, \gamma^{\ell+\delta-1}$ where ℓ, δ are some non-negative integers less than $n-1$. Let $\mathbf{C}$ be the cyclic code with g(x) as the generator polynomial. Then we have*

$$min\{Hamming\ weight(a) : a \in \mathbf{C}, a \neq 0\} \geq \delta$$

Proof. Suppose that the proposition is false. Then there is a non-zero polynomial

$$c(x) = \sum_{i_j} c_{i_j} x^{i_j}$$

with s terms where $s < \delta$ such that $c(\gamma^\ell) = \cdots = c(\gamma^{\ell+\delta-1}) = 0$. Then we the following system of equations

$$\sum_{i_j} c_{i_j} \gamma^{\ell i_j} = 0$$
$$\cdots\cdots$$
$$\sum_{i_j} c_{i_j} \gamma^{(\ell+\delta-1) i_j} = 0$$

Among the above δ linear equations in fewer than δ variables c_{i_j}, we pick the first s so that the number of equations matches the number of variables. The coefficient matrix is the following

$$N = \begin{pmatrix} \gamma^{\ell i_1} & \gamma^{(\ell)i_2} & ... & \gamma^{\ell i_s} \\ \gamma^{(\ell+1)i_1} & \gamma^{(\ell+1)i_2} & ... & \gamma^{(\ell+1)i_s} \\ ... & ... & ... & ... \\ \gamma^{(\ell+s-1)i_1} & \gamma^{(\ell+s-1)i_2} & ... & \gamma^{(\ell+s-1)i_s} \end{pmatrix}$$

It suffices to show that the matrix N is non-singular, then all c_{i_j} must be all zeroes, that implies $c(x)$ is the zero polynomial. Contradiction!

Let us show that the matrix N is non-singular. Let us pull $\gamma^{\ell i_j}$ from the j-th column. Then we have the following matrix L,

$$L = \begin{pmatrix} 1 & 1 & ... & 1 \\ \gamma^{i_1} & \gamma^{i_2} & ... & \gamma^{i_s} \\ ... & ... & ... & ... \\ \gamma^{i_1(s-1)} & \gamma^{i_2(s-1)} & ... & \gamma^{i_s(s-1)} \end{pmatrix}$$

which is a Verdermonde matrix of rank s with x_j replaced by γ^{i_j}. Since we have $\gamma^{i_j} - \gamma^{i_k} \neq 0$ with $i_j \neq i_k \leq n-1$, then the matrices L and N are non-singular. ■

The following cyclic codes were discovered by R.C. Bose and D.K. Ray-Chaudhuri (1960) and A.Hocquenghem (1959) and known as BCH codes.

Definition : Let γ be a *primitive* n-th root of unity in Ω. Let g(x) be the least common multiple of all monic irreducible polynomials of $\gamma^\ell, \gamma^{\ell+1}, \cdots, \gamma^{\ell+\delta-1}$ where ℓ, δ are some non-negative integers less than $n-1$. Then the cyclic code $\mathbf{C}$ with g(x) as the generator polynomial in $\mathbf{F}_q[x]/(x^n-1)$ is called the BCH code of designed distance δ and length n. Usually we take $\ell = 1$. If $n = q^s - 1$ (i.e., γ is a primitive element in $\mathbf{F}_{q^s}$), then it is called a primitive BCH code. ■

We have the following corollary of the preceding proposition.

Corollary 3.1.3 *: The Hamming distance of a BCH code is at least δ.*

Proof. It follows from the preceding **Proposition**. ■

In general it is hard to find the Hamming distance of a BCH code.

Example 9: Let us consider a BCH code $\mathbf{C}$ over $\mathbf{F}_{2^4}$. Let us consider α, β where α satisfies $x^2+x+1=0$ over $\mathbf{F}_2$ and β satisfies $x^2+x+\alpha=0$ over $\mathbf{F}_{2^2}$. It is not hard to see that $\mathbf{F}_{2^2} = \mathbf{F}_2[\alpha]$ and $\mathbf{F}_{2^4} = \mathbf{F}_{2^2}[\beta]$. Let $\gamma = \beta$ satisfying the minimal equation $x^4+x+1=0$ over $\mathbf{F}_2$. Then certainly $\gamma^{15}-1=0$ and γ is a primitive 15−th root of unity. Let us consider the BCH code determined by $\gamma, \gamma^2, \gamma^3, \gamma^4, \gamma^5$. Note that $\delta = 5$ in this case. By checking, one can conclude that $\gamma, \gamma^2, \gamma^4, \gamma^5$ satisfy the following equation,

$$1 + x + x^3 + x^4 + x^5 + x^7 + x^8 = 0$$

and γ^3 satisfies the following equation,

$$1 + x + x^2 + x^3 + x^4 = 0$$

So the generator polynomial is the product of the above two polynomials which is

$$\begin{aligned}&(1+x+x^3+x^4+x^5+x^7+x^8)(1+x+x^2+x^3+x^4)\\&\quad=1+x^3+x^6+x^9+x^{12}\end{aligned}$$

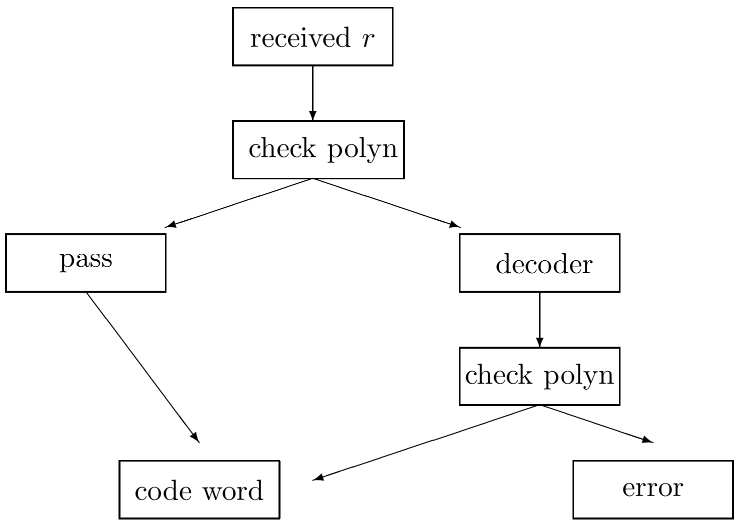
and the check polynomial is $(1+x)(1+x+x^2)=1+x^3$. It is easy to see that the generator polynomial and the check polynomial multiply to $x^{15}-1=x^{15}+1$. ■

3.2 Decoding A Primitive BCH Code

For a useful code, the decoding process is significant. Let us consider a primitive BCH code of length n over $\mathbf{F}_q$ with designed distance δ. Let $g(x)$ be the generating polynomial and c(x)=$c_0+c_1x+\cdots+c_{n-1}x^{n-1}=f(x)g(x)$ be a code word.

During the transmission of the code word, there might be errors and *erasures*, which means the positions of the erased data are known, while the precise data at those positions are unavailabe. Let the received word be $\bar{r}(x)=\cdots+Ex^i+\cdots$ where E's indicate the erasures. We shall treat the letter E as 0 and define $r(x)=\cdots+0x^i+\cdots$. Let the positions of erasures be a set N which are known. Let the number of erasures be w. Let the error word $e(x)$ be defined by $c(x)=r(x)-e(x)$. Clearly given $r(x)$, then $c(x)$ and $e(x)$ determine each other. In the following *Flowchart*, given $r(x)$ the decorder shall produce $e(x)$ if the number of errors and erasures are within a limit. Otherwise it will return an *error* message.

We have the following flowchart,



Flowchart 1

In the above *flowchart*, the decoder[1] is based on the theory of power series and Euclidean algorithm. Even if the decorder produces an error vector $e(x)$, we have to further test the

[1]We will follow the decoding process of SKHN [32]. Another slightly faster decoding process is Berlekamp algorithm [2] decoder. Please see Appendix D.

assumption by checking $c = r - e$ to see if it is a code word using the *check polynomial*. If it is, then we pass to the next block of message. If it is not, then the decoding fails, and we return an *error* message.

Let us concentrate on the *decoder* part. We assume that there are w erasures and at most v errors. Let

$$e(x) = r(x) - c(x) = e_0 + e_1 x + \cdots + e_{n-1} x^{n-1}$$

be the *error vector*. Note that we assume that there are at most $v+w$ non-zero e_i's. Certainly by *brute force*, we may recover c(x). However, it is too slow. A clever way is as follows. We will introduce (follow Peterson [27]),

Definition : Let M be the set of $\{i : e_i \neq 0\}$, the places where either there is an erasure or an error. We shall write the set M as the disjoint union $M = N \cup L$ with N consisting of w erasures and L consisting of at most v errors. The *error-locator polynomial* $\sigma(x)$ is defined as

$$\sigma(x) = \prod_{i \in M} (1 - \gamma^i x) = \prod_{i \in N} (1 - \gamma^i x) \prod_{i \in L} (1 - \gamma^i x) = \sigma_1(x)\sigma_2(x)$$

where $\sigma_1(x)$ **is a known polynomial of degree** w. Thus $\sigma(x)$ and $\sigma_2(x)$ determines each other. The *error-evaluator polynomial* $\omega(x)$ is defined as

$$\omega(x) = \sum_{i \in M} e_i \gamma^i x \prod_{j \in M \backslash i} (1 - \gamma^j x)$$

■

It is easy to see that $deg(\sigma_1(x)) = w, deg(\sigma_2(x)) \leq v, deg(\sigma(x)) \leq w + v$ and $deg(\omega(x)) \leq w+v$. Note that if $\sigma_2(x), \omega(x)$ are known, then the solutions of $\sigma_2(x)$ will give us $\{\gamma^{-i} : i \in L\}$. We take the inverses of all elements in the preceding set, and we have $\{\gamma^i : i \in L\}$, we have the locations of all errors. Furthermore we have $-\frac{\omega(\gamma^{-i})\gamma^i}{\sigma'(\gamma^{-i})} = e_i$ for all erasure or error position i.

Now we will claim the following **sufficient condition** for decoding

$$2v + w = \delta' < \delta$$

The above defined error-locator polynomial $\sigma(x)$ and the error-evaluator polynomial $\omega(x)$ theoretically exist. The whole point is to find them quickly with only the imformations of $r(x)$ and the numerical condition that

$$2v + w = \delta' < \delta$$

Firstly with the above condition satisfied, we expect to find the rational function

$$\frac{\omega(x)}{\sigma_2(x)}$$

for two reasons: (1), the only factor of $\sigma_2(x)$ is of the form $(1 - \gamma^i x)$ with $i \in L$ which is not a factor of $\omega(x)$, therefore the above expression is in the reduced form of the above rational function. (2), the denominator $\sigma_2(x)$ is determined by $\sigma_2(0) = 1$. We have the following proposition,

Proposition 3.2.1 *: We have the following expression in* $\mathbf{F}_q[[x]]$,

$$ord(\frac{\omega(x)}{\sigma_2(x)} - \sum_{i=1}^{\delta'} r(\gamma^i)x^i) \geq \delta' + 1$$

Proof. We have

$$\begin{aligned}\frac{\omega(x)}{\sigma(x)} &= \sum_{i\in M} \frac{e_i\gamma^i x}{1-\gamma^i x} = \sum_{i\in M} e_i \sum_{j=1}^{\infty}(\gamma^i x)^j \\ &= \sum_{j=1}^{\infty}(\sum_{i\in M} e_i\gamma^{ij})x^j = \sum_{j=1}^{\infty} e(\gamma^j)x^j\end{aligned}$$

Note that $c(\gamma^j) = 0$ for $j = 1, \cdots, \delta'$ and $e(\gamma^j) = r(\gamma^j)$ for $j = 1, \cdots, \delta'$. Therefore, we have,

$$\begin{aligned}&ord(\frac{\omega(x)}{\sigma(x)} - \sum_{j=1}^{\delta'} r(\gamma^j)x^j) \\ &= ord(\sum_{j=1}^{\infty} e(\gamma^j)x^j - \sum_{j=1}^{\delta'} r(\gamma^j)x^j) \\ &\geq \delta' + 1\end{aligned}$$

$\blacksquare$

The above equation in the Proposition, written slightly differently as follows, is named by Berlekamp as the *key equation*

$$(1 + S(x))\sigma(x) \equiv \omega(x) \ mod \ x^{\delta'+1}$$

Let $f_1(x) = \sigma_1(x)\sum_{j=1}^{\delta'} r(\gamma^j)x^j \ mod \ (x^{\delta'+1})$ and with $deg(f_1(x)) < \delta'+1$. Note that $f_1(x)$ is thus **uniquely determined**. The conclusion of the above Proposition can be re-written as

$$\begin{aligned}\omega(x) &= \sigma(x)(\sum_{i=1}^{\delta'} r(\gamma^i)x^i)) + x^{\delta'+1}h(x) \\ &= \sigma_2(x)f_1(x) + x^{\delta'+1}h'(x)\end{aligned}$$

Therefore $\omega(x) \in$ the ideal $(f_1(x), x^{\delta'+1}) \in F_q[x]$. In fact, the *Long algorithm* (cf **Proposition 2.5.3**) will provide an easy way to find the rational function $\frac{\omega(x)}{\sigma_2(x)}$. It turns out to be one of the most useful tool in decoding. We have,

Proposition 3.2.2 *(Sugiyama-Kasahara-Hirasawa-Namakawa): We shall use the notations of* **Proposition 2.5.3**. *Let* $f_1(x)$ *be defined above, and let* $f_2(x) = x^{\delta'+1}$. *Let* u *be determined as* $u = 3$ *if* $n_3 \leq v+w$, *otherwise* u *be determined by* $n_{u-1} > v+w \geq n_u$. *For the case* $u = 3$, *we have*

$$\alpha_3(x)f_1 + \gamma_3(x)f_2(x) = 1f_1(x) + 0f_2(x) = f_3(x)$$

otherwise (for u> 3), the following equation,

$$\alpha_u(x)f_1(x) + \gamma_u(x)f_2(x) = f_u(x) \qquad (1)$$

we have

$$\frac{\omega(x)}{\sigma_2(x)} = \frac{f_u(x)}{\alpha_u(x)}$$

Proof. Note that $deg(\alpha_u(x))(\leq \delta' + 1 - n_{u-1}) \leq v$ in any case. Let $\alpha_u(x) = x^s h(x)$ with $h(0) \neq 0$. Then $s \leq v < 2v + w + 1 = \delta' + 1$ and it follows from the above equation (1) that x^s is a factor of $f_u(x)$. We may factor $\alpha_u(x)$ from Eq (1) above and get an power series expression in $\mathbf{F}_q[[x]]$ for the left hand side of Eq (1). We have

$$ord(\frac{f_u(x)}{\alpha_u(x)} - f_1(x)) \geq 2v + w + 1 - s = (v + w) + v - s + 1$$

It follows from **Proposition 2.7.5** that

$$\frac{\omega(x)}{\sigma_2(x)} = \frac{f_u(x)}{\alpha_u(x)}$$

■

From the above Proposition, we easily find $\frac{\omega(x)}{\sigma(x)} = \frac{\omega(x)}{\sigma_1(x)\sigma_2(x)}$.

The following Example will help us to understand the procedure,

Example 10: Let us continue our study of the **Example** of §3.1. It is easy to see that the *generator polynomial*$= g(x) = 1 + x^3 + x^6 + x^9 + x^{12}$. The number $\delta = 5$, we expect to correct two errors. Let the code word be $c(x) = x + x^2 + x^4 + x^5 + x^7 + x^8 + x^{10} + x^{11} + x^{13} + x^{14}$ and the received word be $r(x) = x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^{10} + x^{11} + x^{13} + x^{14}$. Let us assume that there is no erasion $w = 0$ (i.e., $\sigma_1(x) = 1, \sigma(x) = \sigma_2(x)$) and $v = 2$. We take $\delta' = 4$.

We have the following computations,

$$\begin{array}{ll} r(\gamma) = \gamma^2, & r(\gamma^2) = \gamma + 1 \\ r(\gamma^3) = \gamma, & r(\gamma^4) = \gamma^2 + 1 \end{array}$$

We shall start the *long algorithm* with $(f_1, f_2) = (\gamma^2 x + (\gamma + 1)x^2 + \gamma x^3 + (\gamma^2 + 1)x^4, x^5)$. We have

$$\begin{aligned} 1 \cdot f_1 + 0 \cdot f_2 &= 1 \cdot (\gamma^2 x + (\gamma + 1)x^2 + \gamma x^3 + (\gamma^2 + 1)x^4) + 0 \cdot x^5 \\ &= (\gamma^2 x + (\gamma + 1)x^2 + \gamma x^3 + (\gamma^2 + 1)x^4 = f_3 \\ f_2 + (1 + \gamma^7 x)f_3 &= x^5 + (1 + \gamma^7 x)(\gamma^2 x + (\gamma + 1)x^2 + \gamma x^3 + (\gamma^2 + 1)x^4) \\ &= \gamma^2 x + \gamma^{14} x^2 + \gamma^6 x^3 = f_4 \\ f_3 + \gamma^2 x f_4 &= \gamma^2 x = f_5 \\ (1 + \gamma^2 x + \gamma^2(\gamma^3 + \gamma + 1)x^2)f_1 + \gamma^2 x f_2 &= \gamma^2 x = f_5 \end{aligned}$$

We conclude that

$$\frac{\omega(x)}{\sigma(x)} = \frac{\gamma^2 x}{1+\gamma^2 x + (\gamma^2(\gamma^3+\gamma+1))x^2}$$

In other words, we have

$$\omega(x) = \gamma^2 x$$
$$\sigma(x) = 1+\gamma^2 x + (\gamma^2(\gamma^3+\gamma+1))x^2$$

We find two roots of $\sigma(x)$ are γ^{-3}, γ^{-6} (the simple way is trying all $\sigma(\gamma^{-i})$ which takes at most 15 evaluations).

We have to find e_3, e_6. We calculate $\frac{\omega(x)}{x\sigma'(x)}$ at $x = \gamma^{-3}, \gamma^{-6}$, and both values are 1. Therefore, we find $c(x) = r(x) - e = x + x^2 + x^4 + x^5 + x^7 + x^8 + x^{10} + x^{11} + x^{13} + x^{14}$ is a possible code word. Further test by multiplying with the *check polynomial*$= x^3 + 1$ yields $x + x^2 + x^{16} + x^{17} = (x + x^2)(1 + x^3 + x^6 + x^9 + x^{12}) = (x + x^2)g(x)$. We conclude c is a code word. We find the original message is $x + x^2$. ■

Exercises

(1): Show that a Hamming code is a cyclic code.

(2): Find a BCH code with minimal distance $> \delta$.

(3): Write a computer program to decode Example of § 3.2..

(4): Find an example of received word with more than 3 errors in the Example of § 3.2. and is decoded to a wrong code word.

(5): Find the probability of error that the received word r has more than 2 errors, for the Example of § 3.2. We assume that the probabilty of error at any location is p.

3.3 Reed-Solomon Codes

The BCH codes introduced in the preceding sections are general with the disadvantage of being complicated and hard to use. We will discuss simple cases of them, the Reed-Solomon codes, in this section.

The reader is referred to the beginning of Chapter 1. There we discuss a possible way of coding using polynomial curves over real numbers. The process is as follows, let $[a_0, a_1, \cdots, a_{k-1}]$ be the original message. It determines a polynomial curve $f(x) = \sum_0^{k-1} a_i x^i$ of degree $k-1$ at most. Then we send out $[b_1, b_2, \cdots, b_n] = [f(1), f(2), \cdots, f(k), \cdots, f(n)]$. Assuming that there are at most $\lfloor \frac{n-k}{2} \rfloor$ errors, we may use brute force to decode the received message $[b'_1, b'_2, \cdots, b'_n]$ as follows. Knowing k, n, we may try all subsets of $[b'_1, b'_2, \cdots, b'_n]$ of $k + \lfloor \frac{n-k}{2} \rfloor$ elements to see which one determines a polynomial curve of degree at most $k-1$. Once we find the curve, if it exists, we may use it to correct all errors. The difficulty is the decoding process: since a brute force way will be time-consuming. Now we shall change the coefficient field from the real field $\mathbf{R}$ to a finite field $\mathbf{K}$ and rename it the Reed-Solomon code (see below for definition). The important aspect of it is that there is a fast way of decoding (see below).

The reader is further referred to the example in §2.1. which is the starting point of this section. Now we shall use a finite field $\mathbf{F}_q$ in the place of $\mathbf{F}_2$ where $q = 2^m$. We shall use the notation $n = q - 1$. Let γ be a primitive element of the field $\mathbf{F}_q$ (i.e., the powers of γ exhaust all non-zero elements of $\mathbf{F}_q$). We shall consider the points $\{\gamma, \gamma^2, \cdots, \gamma^n\}$ instead of $\{1, 2, \cdots, n\}$ in the real case. Note that $\gamma^n = 1$.

We shall define a code. Let $a = [a_1, a_2, \cdots, a_k]$ be a message. We want to make a code $[c_1, c_2, \cdots, c_n]$. Let $\mathbf{P}_k$ be the set of all polynomials of degree less than k, where $k < n$, over $\mathbf{F}_q$, then $\mathbf{P}_k$ is a vector space of dimension k over $\mathbf{F}_q$. For any vector $[a_1 a_2 \cdots a_k] \in \mathbf{F}_q^k$, let us define $f(x) = \sum a_i x^{i-1}$, and notice that $f(x) \in \mathbf{P}_k$. Let the values at γ^i be $b_i = f(\gamma^i)$ for $i = 1, 2, \cdots, n$. Thus we define a $1 - 1$ map from $a = [a_1 a_2 \cdots a_k]$ to $f(x) = \sum a_i x^{i-1}$, and then to $b = [b_1 b_2 \cdots b_n]$. We have a coding this way.

Proposition 3.3.1 *: Let $k < n$. The set $\{[b_1 b_2 \cdots b_n]$: there is a polynomial $f(x) \in \mathbf{P}_k$ with $b_i = f(\gamma^i)\}$ is a $k-$dimensional subspace $\mathbf{C}$ of the vector space of all n-tuples.*

Proof. Let us define a map π: $\mathbf{P}_k \to \mathbf{F}_q^n$ as $\pi(f(x)) = [f(\gamma), f(\gamma^2), \cdots, f(\gamma^n)]$. Then clearly π is a linear transformation. The only thing we have to show is π is an one-one map, i.e., if $f(\gamma^i) = 0$ for $i = 1, 2, \cdots, n$, then $f = 0$. Note that then $x^n - 1 \mid f(x)$, and deg(f(x))$\geq n$, if $f \neq 0$. A contradiction. ■

We have the following definition,

Definition : The linear code defined by **C** in the preceding proposition is called a **Reed-Solomon code in the value form**, or simply **Reed-Solomon code**. ■

Proposition 3.3.2 *: Let us fix a Reed-Solomon code C. Let b, b' be two distinct vectors in* **C**. *Then $d(b, b') \geq n - k + 1$.*

Proof. . Let $f(x), f'(x)$ define b, b'. Since $b \neq b'$, then $f(x) \neq f'(x)$. Therefore $(f - f')(x)$ has at most $k - 1$ zeroes or at least $n - k + 1$ non-zeroes among $\{1, \gamma, \gamma^2, \cdots, \gamma^{n-1}\}$. ■

Proposition 3.3.3 *: A Reed-Solomon code is a $[n, k, n - k + 1]$-code.*

Proof. . Let f(x)=$\prod_{i=1}^{k-1}(x - \gamma^i)$. Then it is clear that $f(x) \in P_k(x)$ and $f(\gamma^i) = 0$ for $i = 1, \cdots, k - 1$. Let $b_j = f(\gamma^j)$. We have $b_j \neq 0$ for $j = 0, k, k + 1, \cdots, n - 1$. Then the Hamming weight of $[b_1 \cdots b_n]$ is precisely $n - k + 1$. ■

Proposition 3.3.4 *: An Reed-Solomon code in the value form is a MDS code.*

Proof. It follows from the definition of *MDS code* of § 2.4. ■

Remark: Let us assume that $q > 2$ where $q = 2^m$. Although Reed-Solomon code in the value form is a *MDS code* over the field F_q, it is not a *MDS code* over the bit field F_2. Note that if we consider Reed-Solomon code in the value form over the bit field, it will be an [mn,mk] code, with the minimal distance $n - k + 1 \leq (n-k)m + 1$. Furthermore, there is a restriction on the block size by the field size q. Recall that according to Shannon's theorem, to get a sequence of good codes.It is unlikely to have any restriction of the sizes of the blocks. Suppose we keep the rate of information $k/n = \ell$ constant, then the rate of distance $((1 - \ell + 1/n)n)/nm$ will go to zero as $m \to \infty$, i.e., the correcting power will tend to zero, if we allow the size of the field q to get arbitrarily large. Thus the Reed-Solomon-codes in the value form are not the ones forcasted by Shannon's theorem. An additional drawback is that the multiplications over a large field will become more time-consuming. ∎

Another way of defining Reed-Solomon code is to imitate BCH code as follows, note that over $\mathbf{F}_q$, all irreducible polynomials of $\{\gamma^j\}$ are of degree 1,

Definition : A **Reed-Solomon code in the coefficient form**, or simply **Reed-Solomon*** **code**, is a primitive BCH code with length $n = q - 1$. The generator polynomial is of the form $g(x) = \prod_{i=1}^{i=d-1}(x - \gamma^i)$ where $d = n - k + 1$. ∎

Apparently these two codes, Reed-Solomon code and Reed-Solomon* code, are with different encodings. For instance, the code word for $f(x) = 1$ in Reed-Solomon code is $[11 \cdots 1]$ and for Reed-Solomon* code $[\alpha_o \alpha_1 \cdots \alpha_n]$ where α_i is given by the coefficients in the expansion of $1 \times \prod_{i=1}^{i=d-1}(x - \gamma^i)$ and all $\alpha_i = 0 \ \forall i \geq d$. However, we note that the dimension of Reed-Solomon* code is $(n-1) - (d-1) + 1 = k =$ dimension of Reed-Solomon code. In fact, we have the following proposition.

Proposition 3.3.5 *: Let* **C** *be the code space of a Reed-Solomon code and* **C*** *be the code space of the corresponding Reed-Solomon* code. Then* **C**=**C***.

Proof. . We wish to show that **C**$\subset$ **C***. Since they are of the same dimension, then they must be equal. For this purpose, let consider a basis $\{x^j : 0 \leq j \leq k-1\}$ in the message space of **C**. The code word of x^j in **C** is $[\gamma^j \gamma^{2j} \cdots \gamma^{nj}]$. Let us interpret it as the coefficients of the polynomial $f(x) = \sum_{i=1}^{n} \gamma^{ji} x^i = \sum_{i=1}^{n} (\gamma^j x)^i$ (Recall that $x^n = 1$). It is easy to see the following identity,

$$f(\frac{x}{\gamma^j}) = \frac{x^n - 1}{x - 1} = \prod_{i=1}^{i=n-1}(x - \gamma^i)$$

Replacing x by $x\gamma^j$ in the above equation, note that

$$\gamma^{jn} x^n = x^n$$
$$\gamma^j x - 1 = \gamma^j (x - \gamma^{n-j})$$

certainly $n - j \geq n - k + 1 = d$, and we conclude

$$f(x) = \gamma^{-j} \prod_{i \neq (n-j)} (x - \gamma^i) = g(x)h(x)$$

Therefore $f(x) \in (g(x))$.

■

As illustrated by the preceding Proposition, the two different forms of Reed-Solomon codes are equivalent and will be called Reed-Solomon code. The Reed-Solomon code will later be generalized to a *geometric* Goppa code (a type of *algebraic geometric* code). The Reed-Solomon code* can be decoded as an BCH code.

Example 1: Let us consider the following Reed-Solomon* code over $\mathbf{F}_{2^4}$ and $d = 9$. Then we have $q = 16$, $n = 16 - 1 = 15$, $k = 7$. The generator polynomial $g(x)$ is given by

$$g(x) = \prod_{i=1}^{i=8} (x - \gamma^i)$$

where γ satisfies the equation $x^4 + x + 1 = 0$ and is given in the **Example** of §3.1, according to the theory of Reed-Solomon codes, the code may correct up to 4 errors. So $v = 4$ and we assume that there is no erasure, $w = 0$.

Let us consider a received message $r(x) = 1 + \gamma^6 + \gamma x + \gamma^3 x^2 + \gamma^7 x^3 + \gamma^8 x^4 + \gamma^3 x^5 + x^6 + \gamma^3 x^7 + \gamma x^8 + \gamma^2 x^9 + \gamma^3 x^{10} + \gamma^3 x^{11} + \gamma x^{12} + \gamma^2 x^{13} + \gamma^{14} x^{14}$. We will compute $r(\gamma^i)$ as follows,

$$\begin{aligned} r(\gamma) &= \gamma^3, & r(\gamma^2) &= \gamma \\ r(\gamma^3) &= \gamma^7, & r(\gamma^4) &= \gamma^7 \\ r(\gamma^5) &= \gamma, & r(\gamma^6) &= 0 \\ r(\gamma^7) &= \gamma^9, & r(\gamma^8) &= 0 \end{aligned}$$

We will start the long algorithm with $f_1 = \sum_{j=1}^{j=8} r(\gamma^j) x^j = \gamma^3 x + \gamma x^2 + \gamma^7 x^3 + \gamma^7 x^4 + \gamma x^5 + \gamma^9 x^7$ and $f_2 = x^9$ as follows,

$$\begin{aligned} &1 \cdot f_1 + 0 \cdot f_2 = f_3 \\ &f_2 + (\gamma^{13} + \gamma^6 x^2) f_3 = \gamma x + \gamma^{14} x^2 + \gamma^6 x^3 + \gamma^{13} x^4 + \gamma^2 x^5 + \gamma^{13} x^6 = f_4 \\ &f_3 + (1 + \gamma^{11} x) f_4 = \gamma^9 x + \gamma^2 x^2 + \gamma x^4 + \gamma^6 x^5 = f_5 \\ &f_4 + (\gamma^9 + \gamma^7 x) f_5 = \gamma^9 x + \gamma^8 x^2 + \gamma^5 x^3 + \gamma^9 x^4 = f_6 \end{aligned}$$

Notice that $deg(f_5) = 5 > v + w = 4 \geq deg(f_6)$. We substitute back and have the following equation,

$$\begin{aligned} &((\gamma^9 + \gamma^7 x) + (\gamma^{13} + \gamma^6 x^2) + (\gamma^9 + \gamma^7 x)(1 + \gamma^{11} x)(\gamma^{13} + \gamma^6 x^2)) f_1 + (1 + (\gamma^9 + \gamma^7 x)(1 + \gamma^{11} x)) f_2 \\ &= f_6 = \gamma^9 x + \gamma^8 x^2 + \gamma^5 x^3 + \gamma^9 x^4 \end{aligned}$$

with

$$\begin{aligned} &\sigma(x) = \alpha_u(x) = (\gamma^9 + \gamma^7 x) + (\gamma^{13} + \gamma^6 x^2) + (\gamma^9 + \gamma^7 x)(1 + \gamma^{11} x)(\gamma^{13} + \gamma^6 x^2) \\ &\omega(x) = f_u(x) = \gamma^9 x + \gamma^8 x^2 + \gamma^5 x^3 + \gamma^9 x^4 \end{aligned}$$

It is easy to see that $\sigma(\gamma^{-5}) = \sigma(\gamma^{-7}) = \sigma(\gamma^{-10}) = \sigma(\gamma^{-11}) = 0$. Therefore the error locations are $5, 7, 10, 11$. Furthermore, we use the the formula that $-\frac{\omega(\gamma^{-i})\gamma^i}{\alpha'(\gamma^{-i})} = e_i$ for error locations i. We find that $e_5 = 1, e_7 = \gamma^2, e_{10} = 1, e_{11} = \gamma^2 + 1$. We conclude that the error polynomial $e(x)$ and the code polynomial $c(x)$ are

$$\begin{aligned} e(x) &= x^5 + \gamma^2 x^7 + x^{10} + (\gamma^2+1)x^{11} \\ c(x) &= r(x) - e(x) \\ &= 1 + \gamma^6 + \gamma x + \gamma^3 x^2 + \gamma^7 x^3 + \gamma^8 x^4 + (\gamma^3+1)x^5 + x^6 + (\gamma^2+\gamma^3)x^7 \\ &\quad + \gamma x^8 + \gamma^2 x^9 + (1+\gamma^3)x^{10} + (\gamma^3+\gamma^2+1)x^{11} + \gamma x^{12} + \gamma^2 x^{13} + \gamma^{14} x^{14} \end{aligned}$$

The corrected $c(x)$ is a code polynomial which can be checked by multiplying with the following check polynomial.

$$\prod_{i=9}^{15}(x-\gamma^i)$$

and show the result is a multiple of $x^{15}+1$ as follows:

$$(1+\gamma+\gamma^3+\gamma^3 x+(\gamma+\gamma^2)x^2+\gamma^3 x^3+(1+\gamma^2+\gamma^3)x^4+(1+\gamma^3)x^5+(1+\gamma^3)x^6)(x^{15}+1).$$

Therefore the original message is $1+\gamma+\gamma^3+\gamma^3 x+(\gamma+\gamma^2)x^2+\gamma^3 x^3+(1+\gamma^2+\gamma^3)x^4+(1+\gamma^3)x^5+(1+\gamma^3)x^6$. ■

3.4 Classical Goppa Codes

Instead of using polynomials, we shall use the rational functions. We will study a class of linear codes, the classical Goppa codes, which are generalizations of BCH codes. It will be clear from our definitions that the *classical Goppa codes* are not cyclic codes.

Let $g(x)$ be any non-constant polynomial $\in \mathbf{F}_q[\mathrm{x}]$. We shall consider the rings $\mathbf{F}_q[\mathrm{x}]$. Let $\gamma \in \mathbf{F}_q$ such that $g(\gamma) \neq 0$. By Euclidean Algorithm, there exist $\alpha(x)$ and $r \neq 0$ such that

$$g(x) = \alpha(x)(x-\gamma) + r$$

in other words, we have $g(\gamma) = r,\ \alpha(x = \frac{g(x)-g(\gamma)}{x-\gamma)}$ and

$$(-r)^{-1}\alpha(x)(x-\gamma) = 1\ mod\ g(x)$$

Namaely, $(x-\gamma)$ has an inverse $(-r)^{-1}\alpha(x)\ mod\ g(x)$. We say that $(x-\gamma)$ is regular in $\mathbf{F}_{q^m}[\mathrm{x}]/(\mathrm{g(x)})$. It is easy to see that in the polynomial ring $\mathbf{F}_q[\mathrm{x}]$, we have

$$\frac{-1}{g(\gamma)}(\frac{g(x)-g(\gamma)}{x-\gamma})(x-\gamma) \equiv 1\ mod\ g(x)$$

Therefore in the total quotient ring, we have

$$\frac{1}{\overline{\overline{x-\gamma}}} = \frac{-1}{g(\gamma)}(\overline{\frac{g(x)-g(\gamma)}{\overline{x-\gamma}}})$$

or we may write

$$\frac{1}{x-\gamma} \equiv \frac{-1}{g(\gamma)}(\frac{g(x)-g(\gamma)}{x-\gamma})\ mod\ g(x)$$

We have the following definition.

Definition : Let $\mathbf{L}$=$\{\gamma_1,\cdots,\gamma_n\}$ $\in\mathbf{F}_q$. We define the *(classical) Goppa code* $\Gamma(\mathbf{L}$, g(x)) with *Goppa polynomial g(x)* where $g(\gamma_i)\neq 0$ for $1\le i\le n$ to be the set of code words $\mathbf{c}$= $[c_1,\cdots,c_n]$ over the alphabet $\mathbf{F}_q$ for which

$$\sum_{i=1}^{n}\frac{c_i}{(x-\gamma_i)} \equiv 0\ mod\ (g(x))$$

■

The following example will show that the BCH codes are special cases of classical Goppa codes.

Example 1: Let the Goppa polynomial $g(x)$ be x^{d-1}, and $\mathbf{L}$=$\{\gamma^{-i}: 1\le i\le n-1\}$, where γ is a primitive (n-1)th root of unity in $\mathbf{F}_q$. Let us consider the resulting classical Goppa code $\Gamma(\mathbf{L},g)$. Note that $g(\gamma^{-i})=(\gamma^{-i})^{d-1}$. We have the following equations all mod (g(x)),

$$\sum_{i=1}^{n-1}\frac{c_i}{(x-\gamma^{-i})} \equiv 0 \Leftrightarrow \sum_{i=1}^{n-1}\frac{(-c_i)}{(\gamma^{-i})^{d-1}}\frac{x^{d-1}-(\gamma^{-i})^{d-1}}{x-\gamma^{-i}} \equiv 0$$
$$\Leftrightarrow \sum_{i=1}^{n-1}\frac{(-c_i)}{((\gamma^{-i})^{d-1})}\sum_{k=0}^{n-2}(\gamma^{-i})^{d-2-k}x^k = 0 \Leftrightarrow \sum_{i=1}^{n-1}(-c_i)\sum_{k=0}^{d-2}(\gamma^{i(k+1)})x^k = 0$$
$$\Leftrightarrow \sum_{k=0}^{d-2}(\sum_{i=1}^{n-1}(-c_i)(\gamma^{i(k+1)}))x^k = 0$$

Since it is a polynomial of degree less that $d-1 = deg\ g(x)$, we must have all coefficients zeroes, i.e.,

$$\sum_{i=1}^{n-1}(-c_i)\gamma^{i(k+1)} = 0, \text{for k=0,..,}n-2$$

Let a polynomial $f(x)=\sum_{i=1}^{n-1}(-c_i)x^i$. Then we have $f(\gamma^{(k+1)})=0$ for $k+1=1,\cdots,d-1$. We conclude that the last condition is precisely the condition for BCH code. ■

It is clear that a classical Goppa code is linear. It is important to find other parameters. We have

Proposition 3.4.1 *Let $g(x)$ be a polynomial of degree s. Then the classical Goppa code $\Gamma(\mathbf{L},g)$ has a rank $= n-s$ and minimum distance $\ge s+1$.*

Proof. Let us follow the definition of classical Goppa code. From the equation,

$$\sum_{i=1}^{n}\frac{c_i}{(x-\gamma_i)} \equiv 0\ mod\ (g(x))$$

we have that the left hand side can be written as $n(x)/d(x)$ with $d(x) = \prod(x - \gamma_i)$ and deg (n(x)) $<$ n. Since $(d(x), g(x)) = 1$, then $c = \{c_1, \cdots, c_n\}$ is a code word $\Leftrightarrow$ $g(x) \mid n(x)$ or $n(x) = g(x)h(x)$ where $h(x)$ is of degree $< n - s$. Since the classical Goppa codes **C** are parametrized by the coefficients of $h(x)$, therefore its rank is $n - s$. On the other hand, if $c = [c_1, \cdots, c_n]$ is a code word with at most s of c_i's non-zero, then

$$\sum_{i=1}^{n} \frac{c_i}{(x - \gamma_i)} \equiv 0 \; mod \; (g(x))$$

can be rewritten as

$$\sum_{i=1}^{n} \frac{c_i}{(x - \gamma_i)} = \sum_{i \in I} \frac{c_i}{(x - \gamma_i)}$$

where I is an index set with at most s elements. It can be written as $n(x)/d(x)$ with $degree(n(x)) < s$ and $g(x) \mid n(x)$. Therefore $n(x) = 0$. Due to the linear independent property of $\frac{1}{(x-\gamma_i)}$, then all $c_i, i \in I$ must be zero. Hence, we conclude that minimum distance $\geq s + 1$.

■

Proposition 3.4.2 *For any n, let $\mathbf{L}_n$=$\mathbf{F}_{q^n}$. Then there is a sequence of classical Goppa codes $\Gamma(\mathbf{L}_n, g_n(x))$ for some monic irreducible polynomials $g_n(x)$ such that $lim_{n \mapsto \infty} R(\Gamma(\mathbf{L}_n, g_n(x))) \geq 1 - H(\delta)$ where R(M) is the rate of information of the code M and δ is any number with $0 \leq \delta \leq \frac{q-1}{q}$. In other words, the sequence of classical Goppa codes meets the Gilbert-Varshamov's bound.*

Proof. Given δ, let $d_n < n$ be a sequence of integers such that

$$lim_{n \mapsto \infty} \frac{d_n}{n} = \delta$$

Let t_n be a sequence of integers such that $t_n \leq \frac{n}{m}$. We shall find an irreducible polynomial $g_n(x)$ of degree t_n such that the minimal Hamming distance of $\Gamma(\mathbf{L}_n, g_n(x))$ is at least d_n. For this purpose, all code words with Hamming weight less than d_n should be excluded from $\Gamma(\mathbf{L}_n, g_n(x))$. Let $(c_1, \cdots, c_n)$ be a such code word, i.e.,

$$\sum_{i=1}^{n} \frac{c_1}{(x - \gamma_i)} \equiv 0 \; mod \; (g_n(x))$$

We may write the above rational function as $\frac{f(x)}{h(x)}$ and the denominator $h(x)$ is coprime to $g_n(x)$, which means $g_n(x) \mid f(x)$. Let us consider a code word $[c_1, \cdots, c_n]$ of weight j. It is clear that deg f(x)$\leq$ $(j - 1)$ and f(x) has at most $\lfloor \frac{j-1}{t_n} \rfloor$ many irreducible factors of degree t_n. The total number of code words of weight j is $\mathbf{C}_j^n (q - 1)^j$. Therefore, the total number of irreducible polynomials which should be excluded is, by **Proposition 2.6.6**,

$$\sum_{j=1}^{d_n - 1} \lfloor \frac{j-1}{t_n} \rfloor \mathbf{C}_j^n (q - 1)^j i < \frac{d_n}{t_n} \mathbf{V}_q(n, d_n - 1)$$

Let us count the total number of irreducible polynomials of degree t_n. It is, by **Proposition 2.6.6**, with t_n the composite number

$$\mathbf{I}_{t_n} > \frac{1}{t_n} q^{mt_n}(1 - q^{-mt_n/2+1})$$

To show our proposition, it suffices to show that asymptotically, we may find d_n, t_n with

$$\frac{d_n}{t_n}\mathbf{V}_q(n, d_n - 1) < \frac{1}{t_n} q^{mt_n}(1 - q^{-mt_n/2+1})$$

Let us take a logarithm with base q and divide by n. Furthermore, let $m \mapsto \infty$, $(n \mapsto \infty)$ and $\frac{d_n}{n} \mapsto \delta$ (note that then $\frac{d_n - 1}{n} \mapsto \delta$). It follows from **Proposition 2.6.6** that we are looking for

$$H_q(\delta) \leq lim_{n \mapsto \infty} \frac{mt_n}{n}$$

We may select t_n to make the above an equality. Therefore we show the existence of the required irreducible polynomial $g_n(x)$. Further note it follows from **Proposition 3.4.1** that the information rate, $\mathrm{R}(\Gamma(\mathbf{L}_n, g_n(x)))$, of the classical Goppa code $\Gamma(\mathbf{L}_n, g_n(x))$ is at least $1 - \frac{mt_n}{n}$, and we prove that

$$lim_{n \mapsto \infty}\ R(\Gamma(\mathbf{L}_n, g_n(x)) \geq 1 - H_q(\delta)$$

■

Decoding classical Goppa codes

Let $g(x)$ be a polynomial of degree $2t$. Then the classical Goppa code $\Gamma(\mathbf{L}, g)$ has a minimal distance $\geq 2t + 1$. We expect to correct t errors. The process of decoding is as follows.

Let $r = [r_1, \cdots, r_n]$ be the received word. Define the *syndrome*, $S_r(x)$, as

$$S_r(x) = \sum_{i=1}^{n} \frac{r_i}{x - \gamma_i}\ mod\ (g(x))$$

If the *syndrome* $S_r(x) = 0$, then r is a code word. If the *syndrome* $S_r(x) \neq 0$, then we will apply the following process to decode. Note that $(x - \gamma_i)$ and $g(x)$ are coprime. Therefore

$$\frac{1}{x - \gamma_i} \equiv h_i(x)\ mod\ (g(x))$$

where $h_i(x)$ is a polynomial of degree less than $2t$. Then the *syndrome*, $S_r(x)$ can be expressed as a polynomial $h(x)$ which is equivalent to $S_r(x)$ mod $(g(x))$ and of degree less than $2t$. Furthermore, let $r = c + e$ where c is a code word and e is the error word, $M = \{i : e_i \neq 0\} =$ the error locations of r with $\mid M \mid \leq t$. Then $S_c(x) = 0$ and

$$S_r(x) = S_e(x) = \sum_{i \in M} \frac{r_i}{x - \gamma_i} = \sum_{i \in M} r_i h_i(x)\ mod\ (g(x))$$

Let $\alpha_r(x) = \prod_{i\in M}(x-\gamma_i)$ be the error locator of r and $\omega_r(x) = \sum_{i\in M} e_i \prod_{j\in M\setminus i}(x-\gamma_j)$ with degree $< t$. Then we have the following *key equation*,

$$\begin{aligned} S_r(x)\alpha_r(x) &= \sum_{i\in M} \frac{r_i}{x-\gamma_i} \prod_{i\in M}(x-\gamma_i) = \sum_{i\in M} \frac{e_i}{x-\gamma_i} \prod_{i\in M}(x-\gamma_i) \\ &= \alpha_r(x)h(x) = \omega_r(x) \ mod\ (g(x)) \end{aligned}$$

Since $S_r(x)$ is equivalent to $h(x)$ mod $(g(x))$, we conclude that $\omega_r(x)$ is in the ideal generated by $h(x)$ and $g(x)$. We will use long algorithm to find $\omega_r(x)$ and $\alpha_r(x)$ in a routine way as follows. Let $f_1(x) = h(x)$ and $f_2(x) = g(x)$. Now note that $deg(h(x)) < 2t = deg(g(x))$. We have

$$\begin{aligned} &1 \cdot f_1(x) + 0 \cdot f_2 = f_3 \\ &\cdots\cdots \\ &\alpha_u f_1(x) + \gamma_u f_2(x) = f_u(x) \\ &\cdots\cdots \end{aligned}$$

where u is the first time that $n_u = deg(f_u(x)) < t$, i.e., $n_{u-1} = deg(f_{u-1}(x)) \geq t$. We have by Proposition 2.5.3 that $deg(\alpha_u(x))(\leq 2t - n_{u-1}) \leq t$. Combining the above equations, we have

$$\alpha_r(x)(\alpha_u(x)f_1(x)) = \alpha_u(x)\omega_r(x) = \alpha_r(x)f_u(x) \ mod\ (g(x))$$

Since the multiples of the polynomials $\alpha_u(x)\omega_r(x)$ and $\alpha_r(x)f_u(x)$ are of degrees $< 2t = deg(g(x))$, we conclude

$$\begin{aligned} &\alpha_u(x)\omega_r(x) = \alpha_r(x)f_u(x) \\ &\frac{f_u(x)}{\alpha_u(x)} = \frac{\omega_r(x)}{\alpha_r(x)} \end{aligned}$$

Now it is a simple matter to finish the decoding process. (1): Throw away the G.C.D. of $\alpha_u(x)$ and $f_u(x)$ and call the resulting polynomials $\alpha_u^*(x), f_u^*(x)$. Furthermore we make $\alpha_u^*(x)$ monic and adjust $f_u^*(x)$ without changing the fraction, we shall still call them $\alpha_u^*(x), f_u^*(x)$, then we have $\alpha_r(x) = \alpha_u^*(x)$ and $\omega_r(x) = f_u^*(x)$. (2): find all roots of $\alpha_r(x)$, we have M. Let γ_i be a root of $\alpha_r(x)$. Compute $\omega_r(\gamma_i)$. Then we have $e_i \prod_{j\in M\setminus i}(\gamma_i - \gamma_j)$, and hence e_i. We correct the word by taking $c = r - e$, and check c to see if it is a code word. If it is, then we are done. If not, then there are more than t errors, and we return an *error* message.

Exercises

(1): Decode Reed-Solomon $[15, 5, 11]$ code.

(2): Decode the classical Goppa code in the Example of § 3.4.

(3): Write a computer program to decode Reed-Solomon [15,7,9]-code.

(4): We have a document to be preserved. The document is of length $70,000$ alphabets (including all blanks, punctuations, etc.). If the decaying rate is 1% for every ten years and if in case that 30% of the alphabets are ruined, then the whole document is unreadable. How long will the document last if it is written plainly or if it is written with Reed-Solomon code [15,7,9]?

3.5 Product Codes

Let C_1, C_2 be two codes with lengths n_1, n_2, ranks k_1, k_2 and the minimal weights d_1, d_2 respectively. We may arrange the data as a $k_1 \times k_2$ matrix as follows,

$$G = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1k_2} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2k_2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{k_11} & a_{k_12} & a_{k_13} & \cdots & a_{k_1k_2} \end{pmatrix}$$

Then we may apply code C_2 to every row and then code C_1 to every column to form the following $n_1 \times n_2$ matrix

$$G = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1k_2} & \cdots & a_{1n_2} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2k_2} & \cdots & a_{2n_2} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{k_11} & a_{k_12} & a_{k_13} & \cdots & a_{k_1k_2} & \cdots & a_{k_1n_2} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n_11} & a_{n_12} & a_{n_13} & \cdots & a_{n_1k_2} & \cdots & a_{n_1n_2} \end{pmatrix}$$

The above is called a *product code*, in symbol $C_1 \times C_2$, of C_1, C_2

We have the following Proposition

Proposition 3.5.1 *The product code $C_1 \times C_2$ is with length $n_1 \times n_2$, rank $k_1 \times k_2$ and the minimal weight $d_1 \times d_2$. Hence it has a rate $(k_1/n_1)(k_2/n_2)$.*

Proof. The statements about length and rank are trivial. Let us prove the statement about the minimal weight d. Each non-zero row must have at least d_2 non-zeroes and there are at least d_1 non-zero columns (otherwise, every row must be all zeroes). Therefore we have $d \geq d_1 \times d_2$.

On the other hand, let $\mathbf{c_1} = [a_1 a_2 \cdots a_{n_1}] \in C_1$ and $\mathbf{c_2} = [b_1 b_2 \cdots b_{n_2}] \in C_2$ be two codewords with the minimal lengths. Let $a_{ij} = b_i a_j$. Then it is not hard to see that all rows and all columns are codewords, and there are $d_1 \times d_2$ non-zeroes. ■

As shown by the preceding Proposition, the rate of information is decreasing for a product code. On the other hand, the data of a product codes is listed linearly as $a_{11} \cdots a_{1n_2} a_{21} \cdots a_{n_1n_2}$, Any burst of errors will erase a few rows (the first and the last erasures may be only parts of rows) of the matrix. Let ℓ_1, ℓ_2 be the number of erasures codes C_1, C_2 can handle. Then the product code $C_1 \times C_2$ can handle $\ell_2 \times n_1 + \ell_1$ erasures. The decoding procesure is effective to protect the data in a CD against scratches or in a book against the damaging of a part of a page.

PART III

ALGEBRAIC GEOMETRY

Chapter 4

Algebraic Geometry

In the last chapter, we discussed the *Reed-Solomon codes*, which are to evaluate a finite dimensional vector space of functions, i.e., the ring of polynomials of degrees $\leq n$ for a fixed n, on the set of all non-zero values of K. We will see that they are equivalent to evaluate $L(D)$ on P^1_K (see below) with $D = nP_\infty$ (see below). We may view the *Reed-Solomon codes* as codes defined over the projective line (see below). We may extend the concept of *Reed-Solomon codes* to the codes over any projective curve (instead of lines). Now we advance to the next level of coding theory, in which we will focus on the *geometric Goppa code*. We need some knowledge of algebraic geometry. a rich and classical subject. We will not try to give a comprehensive description of it in this book. Instead, we will emphasize the useful curve theory and simply discuss it to the extent that the reader will be able to bear the burden with us. Sometimes we have to give the general picture of algebraic geometry. Most proofs will be deferred to the standard books such as Commutative Algebra [14] by Zariski-Samuel, Introduction to the Theory of Algebraic Functions of one Variable [8] by C. Chevalley and Introduction to Algebraic Geometry [11] by D.Mumford. We will give some examples to illustrate the points.

Usually we consider an *algebraically closed* ground field in algebraic geometry, but in the context of coding theory we have to discuss the case that the ground field is *finite*, and hence we have to consider a field which is **not** algebraically closed. In this survey chapter we will assume that the ground field is either *algebraically closed* or the real field $\mathbf{R}$ or *finite* if not stated explicitly otherwise.

4.1 Affine Spaces and Projective Spaces

In the **plane Euclidean geometry** with real axis, we discuss quadratic curves, ellipses, parabolae, hyperbolae. That is the beginning of algebraic geometry. In the general setup, we will: (1) replace the *real number* axis by an axis over any *field*, (2) replace the words *plane* (i.e., 2-dimensional) by any finite dimensional *affine* or *projective* space, (3) replace a single quadratic equation by any system of polynomial equations (of any degrees).

The point (1) above is natural. The point in (2) of increasing the dimension is natural too. Let us consider the extension to *projective spaces*. The usual point set $\{a_1, \cdots, a_n : a_i \in K\}$

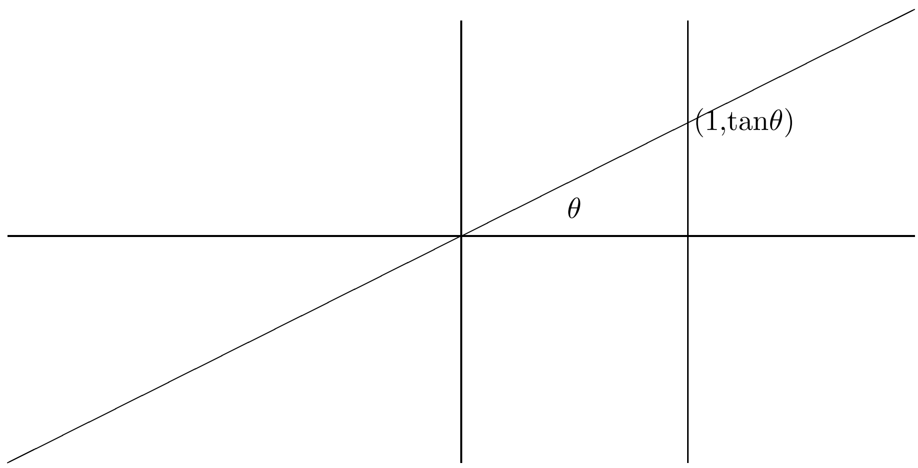
(without the usual vector space structure) is called an n-dimensional *affine space* A^n_K over the field K. There is a serious problem of being *incomplete* for the affine spaces:

Let us consider the intersection of two different lines in the plane. We have that any two different lines will *intersect* at one point "except when they are *parallel*". The last phrase shows that the plane is *incomplete.* We want to say that with points at *infinity*, any two different lines will *intersect* at one point (without exception). The set of all points in affine space A^n_K plus the points at infinity is called n-dimensional *projective space* P^n_K. Rigorously, we define an equivalence relation $\approx$ in the (n+1)-dimensional affine space $A^{n+1}_K \backslash \{0\} = \{(a_0, a_1, \cdots, a_n) : \textit{not all } a_i = 0\}$ as follows,

$$(a_0, a_1, \cdots, a_n) \approx (b_0, b_1, \cdots, b_n) \Leftrightarrow a_i = tb_i \qquad for\ some\ t \neq 0$$

We define the n-dimensional *projective space* P^n_K as $(A^{n+1}_K \backslash \{0\})/\approx$. There is a natural embedding of A^n_K into P^n_K by sending $(a_1, \cdots, a_n)$ to $(1, a_1, \cdots, a_n)$. Let us further explain the point of view.

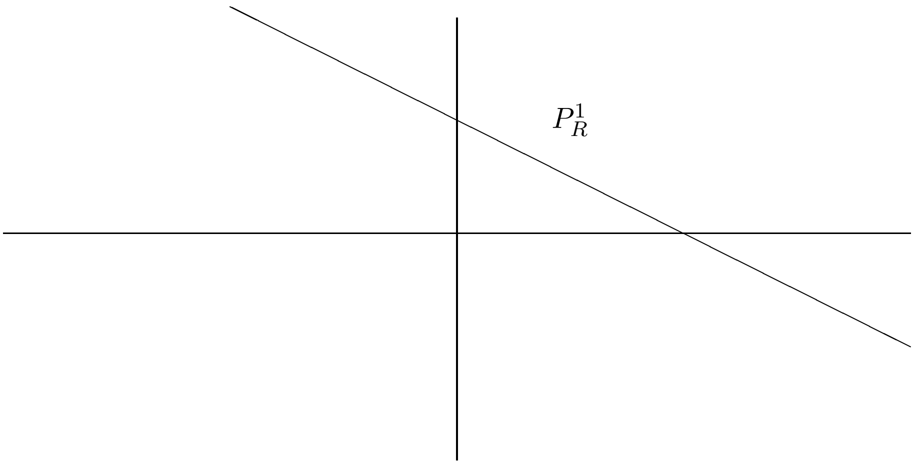
It is easy to see that P^0_K is just the point $\{a_0 : a_0 \neq 0\}/\approx$. For P^1_K, all point can be grouped into two sets: $\{(1, a_1) : a_1 \in K\} \cup (\{(0, a_1 : a_1 \neq 0\}/\approx) i.e., \{(1, a_1) : a_1 \in K\} \cup P^0_K = A^1_K \cup P^0_K$ as in the following picture,



The projective line P^1_K can be represented by all lines passing through the origin. Let the line and the horizental line span an angle θ. As long as $\theta \neq 90°$, the line will have an intersection $(1, tan\theta)$ with the vertical line $(1, t)$, therefore all lines passing through $(0, 0)$ can be represented as the vertical line $x = 1$ union with one extra point (P^0_K or the point at ∞). Note that if the ground field is the real field, then A^1_R is the real line, and P^1_R is a cycle. If the ground field is the complex field, then A^1_C is the complex line which is a real plane, and P^1_C is the one point compactification of the real palne, i.e., a sphere. In general, we have

$$P^{n+1}_K = A^n_K \cup P^n_K.$$

In the case the ground field is the real field, we have the following picture for P^2_R,



Or in the real case we may view it as a unit disc with the antipodes boundary points identified. The interior of the disci is identified with A_R^2 by $v \mapsto v/(1-|v|)$ where v is an vector inside the unit disc, $|v|$ is the length of v.It is a non-rientable surface. Let C be a circle on P_R^2 in this picture. Then it will separate the projective plane into two parts, one of them is a disc, the other one is a Mobius band. We may think that the disc is the interior of the circle and the Mobius band is the exterior of the circle.

From a topological point of view, the affine spaces are not *complete* over the complex numbers C, while the projective spaces are.

We define a *linear subspace* L in P_K^n as the image of $V\backslash\{0\}$ where V is a positive dimensional subspace of the A_k^{n+1} treated as a vector space. We define $dim(L) = dim(V) - 1$.

We have the following well-known theorem about vector space S:

Proposition 4.1.1 *Let V_1, V_2 be two subspaces of a vector space K^{n+1}. Then we have*

$$dim(V_1 \cap V_2) \geq dim(V_1) + dim(V_2) - (n+1)$$

■

Proposition 4.1.2 *Let L_1, L_2 be two linear spaces of P_K^n. Then we have*

$$dim(L_1 \cap L_2) \geq dim(L_1) + dim(L_2) - n$$

■

Corollary 4.1.3 *Let $dim(L_1) = dim(L_2) = 1$ and $n = 2$. Then the intersection of L_1, L_2 is not empty.*

■

Even over a finite field, both A_K^n and P_K^n are finite, hence both are compact. However the distinction pointed out by the preceding Corollary is obvious. We will say that the projective spaces are *complete*. The above Corollary will be generalized to Bezout's Theorem (see below).

4.2 Affine Algebraic Varieties

Although we only worry the usage curve theory for coding theory. We shall enjoy the beautiful theory of *algebraic geometry*. Let us speak further about A_K^n. There are two objects involved: (1) *the algebraic objects* of the affine space A_K^n, including a polynomial ring $K[x_1, \cdots, x_n]$ (over the projective space P_K^n, there is a *homogeneous* polynomial ring $K[x_0, x_1, \cdots, x_n]^h$ ={ all homogeneous polynomials}), and the system of equations there, and (2) *geometric objects* of the affine space A_K^n, which is the solution set $X(I)$ of an ideal of equations I . The relation between these two objects may be complicate in general. Primarily, we take the point of view of algebra.

We have the following well-known Hilbert basis theorem:

Proposition 4.2.1 *(Hilbert basis thereom): The ideal generated by any system of polynomials in n variables can be generated by finitely many polynomials.*

Proof. Zariski-Samuel V I [14]. ■

Note that $X(I)$ may be empty even if $I = (f)$ where $f(x_1, \cdots, x_n) \neq$ non-zero constant. Look at the following examples;

Example 2: Let $K = R$, the real field, and $f(x_1, \cdots, x_n) = x_1^2 + 1$, Then there is no solution. However, if we extend the field R to the complex field C, then there are two solutions $x_1 = i, -i$. ■

Example 3: It is very important to know the number of solutions of equatons for the construction of *algebraic geometric coding*. This is a no-trivial problem. We want to indicate that sometimes there is no solution for a non-trivial equation. Let K be a finite field with q elements where $q = p^m$, and let $x_0^{q-1} + \cdots + x_n^{q-1} = 0$, where $n \leq p - 2$, be a homogeneous polynomial, Since $x_i^{q-1} = 1$ for any $x_i \neq 0$. Then in P_K^n, there is no solution. However, over a suitable finite field extension of K, there will be solutions. ■

If we want to get a 1-1 correspondence between the algebraic objects and the geometric objects, then we have to go to some suitable extensions of the ground field K. If we consider all algebraic objects, as rings, ideals etc., we had better consider the ultimate algebraic field extension, the *algebraic closure* of K. In algebraic geometry, we usually assume that the field K is algebraically closed from the very beginning. However, if we only consider a fixed ideal I, we may be satisfied with some finite field extension of K. The advantage is that if K is a finite field, then a finite extension of it stays finite.

Let us see what happens if we replace the field K by its algebraic closure $\bar{K}$ or simply assume that K is algebraically closed. Then we have the following well-known theorem:

Proposition 4.2.2 *(Hilbert's nullstellensatz): Let K be an algebraically closed field. Let I be an ideal of $K[x_1, \cdots, x_n]$ and S any subset of A^n_K. Let $X(I) = \{(a_1, \cdots, a_n) : (a_1, \cdots, a_n) \in K^n,\ g(a_1, \cdots, a_n) = 0,\ for\ all\ g \in I\}$ and $J(S) = \{g : g \in K[x_1, \cdots, x_n],\ g(a_1, \cdots, a_n) = 0,\ for\ all\ (a_1, \cdots, a_n) \in S\}$. Then*

$$Rad\ I = J(X(I)) = \{f \in K[x_1, \cdots, x_n] : f(a_1, \cdots, a_n) = 0,\ for\ all\ (a_1, \cdots, a_n) \in S\}$$

where $Rad\ I = \{h \in k[x_1, \cdots, x_n] : h^m \in I\ for\ some\ suitable\ m\}$.

Proof. Zariski-Samuel VII, page 164 [14]. ■

Example 3: Let **C** be the field of complex numbers. Note that **C** is algebraically closed. Let I be a proper ideal (i.e., $I \neq (1)$). Then $I \subset$ some maximal ideal M. Let Rad (I)=I' (see the definition of Rad in the previous Proposition). We have $\mathbf{C}[x_1, \ldots, x_n]/M \equiv \mathbf{C}$. Let $(\bar{x}_1, \cdots, \bar{x}_n) = (a_1, \cdots, a_n)$. Then clearly $M = (x_1 - a_1, \cdots, x_n - a_n)$, and $f(a_1, \cdots, a_n) = 0$ for all $f \in I$. In other words, the ideal I' has at least one common solution. ■

If we have an algebraically closed ground field, then we define an *algebraic variety* to be the solution set of an ideal I which equals *Rad* I. We define a *radical ideal* I to be an ideal I which equals *Rad* I. Then the above theorem implies that in the case of algebraically closed ground fields K, there is a natural 1-1 correspondence between the set of algebraic varieties (=the sets of solutions of system of equations from radical ideals) and the set of radical ideals. In other words, in a proper set-up, the *algebraic objects* and the *geometric objects* reflect each other.

However, if we have a non-algebraically closed ground field, then we can not establish the above correspondence. Let us consider the case that K is a finite field (a finite field is never algebraically closed). Two different radical ideals may have the same solution sets. We may (1) find a finite extension of K to distinguish these two ideals. If we go to the algebraic closure $\bar{K}$ of K, then the solution sets must be distinct. Therefore, there are points belong to one of the two solution sets while not the other one. Let us pick any one of the points, say $(a_1, \cdots, a_n)$, then $a_i \in \bar{K}$, hence algebraic over K, therefore *finite algebraic* over K. If we join all a_i to K, we get a finite extension K' of K and $(a_1, \cdots, a_n)$ belong to one of the solution sets while not the other. We consider these points gotten by finite extensions *virtual points* or *point classes*. Implicitly, when we consider a radical ideal I, we consider all points and *virtual points*. This is one approach to the problem. (2) to avoid the problem of field extensions, in any case, we will redefine the point set or the **variety** of I as the collection of all maximal ideals containing I, i.e., $X(I) = \{m : m\ maximal\ ideal \supset I\}$ =m-spec(R/I)=JacRad(I). What is needed algebraically is to prove $JacRad(I) = \cap_{p \supset I}\ p = Rad(I)$ where p's are prime ideals. Any ring with the property that $JacRad(I) = Rad(I)$ for all ideals I will be called a *Hilbert's* ring (or *Jacobson's* ring). It is known that if R is a *Hilbert* ring then $R[x]$ and any homomorphic image is a *Hilbert* ring. It is easy to see that Z and any field K is a *Hilbert* ring. Then any $Z[x_1, \cdots, x_n]$, $K[x_1, \cdots, x_n]$ are *Hilbert* rings.

Example 4: Let us consider the ring Z of all integers. Let $I = (0)$. Then clearly all maximal ideals are of the form (p) where p is a prime number. We have $X((0)) = \{(p) : p\ prime\}$. And $X((p)) = (p)$, $X((12)) = \{(2), (3)\}$. Moreover we have $Rad((12)) = (2) \cap (3) = (6)$.

■

Exercises

(1): Show that the power series ring $K[[x]]$ is not a *Hilbert's* ring.

(2): Let $f = x^2y^3 \in K[x, y]$. What is $Rad\ (f)$?

(3): Show that $Rad((x^2 + 1)) \neq Rad((1))$ in the ring $R[x]$ where R is the field of real numbers.

(4): Find a finite set of generators for the ideal generated by $\{x^i - y^{i+1} : i = 10, 11, \cdots\}$ in the polynomial ring $K[x, y]$ of two variables.

(5): Is the ideal $(x^1 + 1, y^2 + 1) \in R[x, y]$ prime? radical? where R is the field of real numbers.

4.3 Regular Functions and Rational Functions

One of the central approaches of mathematics is to define a set of functions $R(S)$ on a set S, and to study the relations between $R(S)$ and S. We may have *differentiable* or *analytic* functions on a set S, then we may have differentiable or analytic geometry. In algebraic geometry, we are more interested in polynomials or rational functions. In our present situation, we will define *regular functions* as polynomials. Given an affine algebraic variety X defined by an ideal $J(X)$, we define the *ring of regular functions* $R[X]$ as $K[x_1, \cdots, x_n]/J(X)$ $=K[\bar{x_1}, \cdots, \bar{x_n}]$, i.e., we only consider *induced polynomials* from the affine space A^n_K to X. The value of a polynomial f at a maximal ideal m is defined to be the image of f under the canonical map to $R[X]/m$. Naturally, two induced polynomials are considered to be identical iff they have identical values on the algebraic variety (virtual points included) iff the values on m-spec are identical.

Let us redefine *points* algebraically as follows:

Definition : A point P of a given algebraic variety X over a field K with the ring of regular functions $R[X]$ is a maximal ideal of $R[X]$. If $K \equiv R[X]/P$, then the point is called a *rational point.*

■

Example 1: Let X be the affine line A^1_R over real field R. Then $(x^2 + 1)$ is a maximal ideal of $R[x]$. It is a point. However $x^2 + 1 = 0$ has no real solution and $R[x]/(x^2 + 1) \equiv C \not\equiv R$ where C is the field of complex numbers. So the point $(x^2 + 1)$ is not a *rational point.* ■

Definition : Let P be a maximal ideal. The algebraic degree of field extension $[R[X]/P : K]$ is called the *residue degree* $\mu(P)$. Rational points are point with residue degree 1. ∎

We have the following Proposition.

Proposition 4.3.1 *: A point P is rational $\Leftrightarrow$ $P = (x_1 - a_1, \cdots, x_n - a_n)$ where $a_i \in K$ for all i.*

Proof. ($\Rightarrow$) If P is rational, then $R[X]/P = K$, so $\bar{x_i} = a_i \in K$ for all i. Furthermore, $(x_1 - a_1, \cdots, x_n - a_n)$ is a maximal ideal, and $P = (x_1 - a_1, \cdots, x_n - a_n)$.
($\Leftarrow$) If $P = (x_1 - a_1, \cdots, x_n - a_n)$, then $R[X]/P = K$. ∎

For simplicity, we will consider the case that $K[x_1, \cdots, x_n]/J(X)$ is an *integral domain*, i.e., $J(X)$ is a *prime ideal*. Then the algebraic variety X will be called an *irreducible algebraic variety*. We will restrict our attention to only the set of irreducible algebraic varieties, and by abusing the language, using the term *algebraic variety* only for an *irreducible algebraic variety*, if not stated explicitly otherwise. In this case, the *total quotient ring* of $R[X]$ is a field. We will call it the *rational function field* $F(X)$ of the algebraic variety X. If the ideal $J(X)$ stays prime for all field extensions of K, then we say the algebraic variety X is *absolutely irreducible*. The ideal $(x^2 + 1)$ of $R[x]$ is not absolutely irreducible.

Definition : We define the *local ring* of an affine algebraic variety X at a point P as $K[\bar{x_1}, \cdots, \bar{x_n}]_p = \{f/g : f \in K[\bar{x_1}, \cdots, \bar{x_n}],\ g \notin P\}$. Algebraically, we define a *local ring*, (R, q), as a ring R with a unique maximal ideal q. It is easy to see that the local ring (R, q) of an affine algebraic variety X at a point P is a local ring in the sense of algebra. We recall the definition of its $q-$adic completion $\hat{\mathrm{R}}$ as the completion of R with respect to the metric induced by the ideal q as follows. In general let (R, q) be a local ring, we define the order distance function as

$$ord_q(f) = max\{m : f \in q^m\}$$

if $f \in F(X)\backslash 0$, otherwise ∞. Furthermore we define the distance between $f, g \in R$ as

$$d(f, g) = 2^{-ord_q(f-g)}$$

It is routine to check that R forms a metric space with respect to the preceding defined distance function. ∎

We have a *dimension* theory of local rings. It is the maximal length of chains of prime ideals. The reader is advised to consult Zariski-Samuel [14] about the definition.

Definition : A local ring (R, q) of *dimension* m is said to be a *regular* local ring iff its maximal ideal q can be generated by m elements $\{x_1, \cdots, x_m\}$ which are called *uniformization parameters*. In the equicharacteristic (the residue field R/q and the quotient field of R has the same characteristic. cf Zariski-Samuel V II page 304 [14] *Cohen's theorem*) its completion $\hat{\mathrm{R}}$ is a formal power series ring in m variables. In this case, $\hat{\mathrm{R}}$= $\bar{K}[[x_1, \cdots, x_m]]$ where $\bar{K}$ is algebraic over K, and the point is *rational* iff $\bar{K} = K$. A point P is said to be a regular point if the *local ring* of an affine algebraic variety X at P is regular. ■

Example 2: Let p be a prime number. Then the local ring Z_p is regular while its residue field is of characteristic $p > 0$. Therefore it is *not* equicharacteristic case. Its completion is the $p-$adic numbers which is not a formal power series ring. ■

Example 3: Let $P = (x^2 + 1)$ be a maximal ideal in $\mathbf{R}[x]$ where $\mathbf{R}$ is the real number field. Then the local ring of $\mathbf{R}[x]$ at P is regular with $\mathbf{R}[x]/(x^2 + 1) \equiv \mathbf{C}$ where $\mathbf{C}$ is the field of complex numbers. In the completion of $\mathbf{R}[x]_{(x^2+1)}$, let $t = x^2 + 1$, then there is an element

$$\frac{x}{\sqrt{1-t}} = i$$

the completion of $\mathbf{R}[x]_{(x^2+1)}$ is $\mathbf{C}[[t]]$. ■

Exercises

(1): Let us consider a curve over a field K defined by

$$xy + x^3 + y^3 = 0$$

Is the local ring of C at $(0,0)$ regular?

(2): Let R be the field of real numbers. Show that there are no rational points for two varieties X_1, X_2 defined by ideals $(x^2 + 1)$ and (1). Furthermore, show that there are non-rational points in X_1 but not X_2.

(3): Prove Eisenstein's criterion: Let $f(x, y) \in K[x, y]$ be written as,

$$f(x,y) = \sum_{i=0}^{n} f_i(x) y^i$$

Suppose that $(f_0(x), f_1(x), \cdots, f_n(x)) = (1)$ and that there exists $b \in L$ which is a finite extension of K such that

$$f_n(b) \neq 0$$
$$f_i(b) = 0, \; for \; i < n$$
$$(x-b)^2 \dagger f_0(x)$$

Then $f(x, y)$ is absolutely irreducible.

(4): Using the above Eisenstein's criterion to show that $y^n - x$ is absolutely irreducible.

(5): Let P_1 be the point $(0,0)$ on P^1_K with the local ring O_{P_1}. Let P_2 be the point $(0,1)$ on the curve defined by $x^3 + y^3 = 1$ with the local ring O'_{P_2}. Show that O_{P_1} is non-isomorphic to O'_{P_2}. Show that $\hat{O}_{P_1}$ is isomorphic to $\hat{O}'_{P_2}$.

(6): Let $P = (x^2+1)$ be a maximal ideal in $\mathbf{R}[x]$ where $\mathbf{R}$ is the real number field. Show that the local ring of $\mathbf{R}[x]$ at P is regular.

4.4 Affine Algebraic Curves

In coding theory, the only part of *algebraic geometry* we will use is the *algebraic curve theory.* We have the following theorem.

Proposition 4.4.1 *(Noether's Normalization Lemma): Let R be an integral domain which is a finitely generated extension ring of a field K. Let r be the transcendental degree of the quotient field F of R over K. Then there is an algebraically independent subset $\{t_1, \cdots, t_r\}$ such that R is integral over $K[t_1, \cdots, t_r]$.*

Proof. Zariski-Samuel VI page 104 [14]. ∎

Dimension theory is an interesting subject. For linear algebra, we define the *dimension* of a vector space as the number of elements of a basis. In topology, some times we use the *Cěch* theory of dimension. In analysis, we have Hausdorff dimension. In algebraic geometry, we define the *dimension* of A_K^N (or P_K^n) to be n, and we can show that the dimension is invariant under an integral extension. The above lemma lays the ground for us to define the *dimension* of an (irreducible affine) algebraic variety X to be the *transcendental degree* of the field $F(X)$ over the field K. A *curve is an (irreducible affine) algebraic variety X of dimension* 1.

We have the following definition of non-singularity by the Jacobian criterion,

Definition : A rational point $x = (a_1, \cdots, a_N)$ is a non-singular point of a variety $X = X(K[x_1, \cdots, x_N]/\ (f_1, \cdots, f_n)))$ of dimension n iff

$$rank[b_{ij}] = N - n$$

where

$$b_{ij} = \frac{\partial f_i}{\partial x_j}(a_1, \cdots, a_N)]$$

Otherwise it is called a *singular* point. ∎

Proposition 4.4.2 *: For our case of a perfect ground field, a point is regular iff it is non-singular.*

Proof. Mumford page 343. ∎

Proposition 4.4.3 *: Let $f(x, y) = 0$ define an affine plane curve C. Then the collection of all singular points, the singular locus, is defined by the ideal $(f(x, y), f_x, f_y)$.*

Proof. It follows from the definition. ■

Proposition 4.4.4 *: Let $f(x,y) = 0$ define an absolutely irreducible affine plane curve C. Then the affine plane curve C is non-singular iff $(f(x,y), f_x, f_y) = (1)$.*

Proof. It follows from the previous Proposition. ■

Example 1: Let n be a positive integer such that $p \dagger n$. Let us consider the following Fermat curves over $\mathbf{F}_{p^m}$ with equations,

$$x^n + y^n - 1 = 0$$

It follows from the above Proposition that it is non-singular for this affine part. ■

In general, we may discuss a non-algebraic curve cases, even *arithmetic* cases. See the following example.

Example 2: Let us consider the ring Z of all integers. Let us the term *spec* to denote the collection of all prime ideals. For Z, its $spec = \{(0)\} \cup \{p : p\ prime\}$. The field Q of rational numbers is the quotient of Z_0, the localization of Z at the prime ideal 0, which can be considered as all rational functions. Let a/b be a rational number (function) in the reduced form. If $p \dagger b$, then $a/b\ mod(p) \in Z/pZ$ can be considered as its *value* at the point (p). If $p \mid b$, then its value is ∞. ■

The algebraic closure of K in the quotient field of R, is called the *field of constants*. For simplicity, we will assume that $K =$ the field of constants.

Proposition 4.4.5 *: Let F be a finitely generated field extension of a field K. If the ground field K is perfect, then there is a transcendental basis $\{x_1, \cdots, x_n\}$ such that F is separable over $K(x_1, \cdots, x_n)$.*

Proof. Zariski-Samuel VI page 105 [14]. ■

In case that the transcendental degree $n = 1$, since every finite separable extension is a simple extension, if the field K is perfect, we may reduce the field to $K(x, y)$. In other words, we find a plane model for the curve. Although any curve has a plane curve as a model (those two curves share the same rational function field, or those two curves are *birationally equivalent* (see below for *definition*)), a plane curve will have at most $\frac{q^3-1}{q-1} = q^2 + q + 1$ rational points (P_K^2 has so many points). A non-plane curve may have many more rational non-singular points. If the number of points matters in our discussion (see the next chapter on *geometric Goppa codes*), then we may not restrict all curves to planal curves.

Let C be a curve, recall that the curve is said to irreducible if the ideal $J(C)$ is prime. Let us consider the following example:

Example 3: Let the field $K = R$ the field of real numbers. Let $J(C) = (x^2 + 1)$ in $R[x, y]$. Then $J(C)$ is irreducible. However if we extend the field R to the complex field, then the generating polynomial (x^2+1) splits into a product $(x+i)(x-i)$. So the curve splits too. ∎

Recall that a curve is said to be an *absolutely irreducible curve* if it stays irreducible after any finite ground field extension (or go to the algebraic closure of K, it is the same thing).

Definition : Let F, G be two not necessarily irreducible plane curves defined by equations $F = 0, G = 0$ respectively, and P be a point of A^2_K (i.e., a maximal ideal in K[x,y]). Let O_P be the local ring of A^2_K at P. Then the *intersection number* of F, G at P, $I(P, F \cap G)$, is defined to be $dim_K(O_P/(F,G)) = \ell(O_P/(F,G))\mu(P)$ where $\mu(P)$ is the residue degree (cf Definition of sec 4.3). ∎

Example 4: Let $K = R$ the real field. Let F, G be the following system of equations,

$$F = x - 2$$
$$G = x^2 + y^2 - 1$$

Then it is easy to see that $P = (x-2, y^2+3) = (F,G)$ is a maximal ideal. $\ell(O_P/(F,G)) = 1$ and $\mu(P) = 2$. On the other hand, if we consider the same set of equations over C, the complex field and $P_1 = (x-2, y-\sqrt{3}i)$, and $P_2 = (x-2, y+\sqrt{3}i)$. Then we have $\ell(O_{P_j}/(F,G)) = \mu(P_j) = 1$ for $j = 1, 2$. ∎

In the next Proposition, we will use the classic *Resultant* or *Sylvester Resultant*. The reader is advised to consult any book on algebra (see for instance, Van der Waerden *Modern algebra*).

Proposition 4.4.6 : *We always have $I(P, F \cap G) = ord_P(Resultant_x(F,G))$.*

∎

Example 5: Let $F = x^2 + y^3$ and $G = y^2 + x^3$. Then

$$dim_K(O_P(F,G)) = 4 = ord_P(Resultant_x(F,G))$$

∎

Exercises

(1): Let C be a curve defined over K with $\mathrm{ch}(K) = 3$ by the following equation

$$xy + x^3 + y^3 = 0$$

find the local rings at $(0,0)$ and $(1,1)$ and their completions.

(2): Find the pole set of x for the curve P_K^1 with regular function ring $K[x]$.
(3): Find $I(P, F \cap G)$ where $P = (0,0)$, $F = x + y^3$, $G = x + a$ on the palne.
(4): Show that $x + y^n$ is absolutely irreducible.
(5): Show that the local rings of $x^3 + y^3 = 1$ at $(1,0)$ and the line x at (0) are not isomorphic, while their completions are.

4.5 Projective Algebraic Curves

We defined a *projective algebraic variety* to be an affine algebraic variety with points at infinity added. To be precise, with an ideal I in $K[x_1, \cdots, x_n]$, we define the homogeneous ideal I^h associated with it as follows,

$$I^h = \{x_0^d f(x_1/x_0, \cdots, x_n/x_0) : f \in I, d = deg(f)\}$$

and we define the projective completion as $X(I^h)$. Then we define *projective variety* as the projective completion of an affine algebraic variety. Note that the affine piece defined by $x_0 = 1$ is the original affine variety. Similarly, we may take one of $x_1, \cdots, x_n$ to be 1, then we get the other n affine parts of the projective variety.

Example 1: Let n be a positive integer such that $p \dagger n$. Let us consider the following *projective* Fermat curves over $\mathbf{F}_{p^m}$ with equations,

$$x_1^n + x_2^n = x_0^n$$

Let us consider the affine part defined by $x_0 = 1$. Let $x = x_1/x_0$, $y = x_2/x_0$. Then the above can be rewritten as

$$x^n + y^n = 1$$

In the discussion of **Example 4** of section 5.3, we know it is non-singular for this affine part. Similarly, for other affine parts, it is non-singular. So it is non-singular as a projective curve.

■

We define the rational functions of an algebraic variety (affine or projective) as elements in the rational function field $F(X)$. Note that there is a major difference between the concepts of *functions* as in the rational function field and *functions* of Calculus (say C[0,1], the set of continuous functions defined over [0,1]). Usually, in the second case, functions have a common domain of definition.

The common domain of definition of all rational functions is the empty set. For the rational function field $F(X)$, there are functions which have a pole at any given rational point P, say, let f be a non-constant rational function not having a pole at P, $f(P) = a$, then $1/(f(x) - a)$ will have a pole at P. Therefore the common domain of definition is the empty set, not the algebraic variety X. This is a point which is taken up by the *sheaf theory* and makes algebraic geometry interesting.

We wish to generalize the Corollary 4.1.3 on the intersection of two lines on P_K^2. Let us consider any two forms F, G of degrees m, n respectively. They define two not necessarily irreducible projective plane algebraic curves C, D of degree m, n respectively. We say that they have a component if F, G have a non-constant common factor. Let us quote the important Bezout's Theorem as follows,

Proposition 4.5.1 *(Bezout's Theorem): Let* **C, D** *be two not necessarily irreducible projective plane algebraic curves of degree m, n respectively. If they do not have a common component, then they intersect at mn points with intersection numbers (cf Definition of section* 4.4*) counted.*

Proof. . This theorem follows from the computations of resultant. ∎

Example 1: Let us consider two curves $x^2 + x + 1$, $y^2 + y + 1$ defined over a field K with characteristic 2. We compute the intersection numbers for two cases, (1) the ground field $K \supset \mathbf{F_{2^2}}$, (2) the ground field $K \not\supset \mathbf{F_{2^2}}$.

(1): Clearly each of the two curves dissolves into two lines, and the intersection number is $4 = mn$.

(2): The equation $y^2+y+1 = 0$ can be replaced by $y^2+y+1+x^2+x+1 = (x+y)(x+y+1) = 0$. We have two points $P_1 = (x^2 + x + 1, y + x), P_2 = (x^2 + x + 1, y + x + 1)$ in their common intersection. At each point, we have,

$$O_{P_i}/(F, G) \approx K + Kx$$

Hence we have $\sum I(P, F \cap G) = 4 = mn$. ∎

In general we define the *divisor group*[1], $D(C)$, of an non-singular projective algebraic curve C as the abelian group generated by all points P_i (rational or not). We define

Definition : Let D be in $D(C)$. Then D is called a *divisor*. Let

$$D = \sum_i n_i P_i$$

Then the degree of **D**, d(**D**), is defined to be,

$$d(\mathbf{D}) = \sum_i n_i \mu(P_i).$$

where $\mu(\mathbf{P_i})$ = residue field degree=$[R(C) : K]$. We use the following notation,

$$ord_{P_i}(\mathbf{D}) = n_i = (\mathbf{D}(P_i))$$

∎

[1] Chevalley [8] uses multiplicative formate of divisor group as against additive group formate of divisor group in this book

For a rational function f, let P_i be any point of C (note that all points are regular), we define $ord_{P_i}(f)$ to be the order of f at the local ring O_{p_i}, we define (f), the divisor of f as

$$(f) = \sum_{P_i} ord_{P_i}(f) P_i$$

Example 2: Let us consider the real field and the projective line P^1_R over R. Let us consider the function $f = x^2 + 1$. It vanishes at the maximal ideal $P = (x^2 + 1)$ wth residue field degree $\mu_p(f) = 2$, and a double pole at ∞ P_∞. Therefore, it's divivor $D = (f) = P + 2P_\infty$ and degree $d((f)) = 1 \cdot 2 - 2 \cdot 1 = 0$. ■

Proposition 4.5.2 : *Let f be any non-zero rational function. Then (f) is a divisor of degree 0. In other words, any non-zero rational function has the same finite number of zeroes as poles, if they are counted properly.*

Proof. Any rational function can be considered as the quotient of two homogeneous polynomials of the same degree. Let us take a plane model, then this proposition follows from Bezout's Theorem. ■

The following is a genralization of classical *Liouville's theorem*,

Proposition 4.5.3 : *Every non-constant function f has some poles.*

Proof. We will prove only for a plane model. It is easy to see that if for all affine parts, f is constant, then f must be constant. Let us consider that for one affine part C, f is not constant. Since the divisor (f) is of degree 0, then f has some poles $\Leftrightarrow$ f has some zeroes. note that $F(C) = \{\bar{g}/\bar{h} : \bar{h} \neq 0, \bar{g}, \bar{h} \in R[C]\}$. Let us assume that $f = \bar{g}/\bar{h}$ with g/h in its reduced form and not a constant. If h is not a constant, then let $p(x, y)$ be any prime factor of $h(x, y)$, clearly $p(x, y)$ is not a factor of g, and $(\bar{p}(\bar{x}, \bar{y}))$ will be a pole of f. If h is a constant, we may take it to be 1. In this case g can not be a constant. Let $q(x, y)$ be a prime factor of g. Then $(\bar{q}(\bar{x}, \bar{y}))$ will be a zero of f. ■

Definition : A divisor $D = \sum_i n_i P_i$ is said to be effective if all n_i's are non-negative. Let f be a rational function with the divisor $(f) = \sum_i n_i P_i$. Then the partial sum with $n_i > 0$ will be called the zero-divisor of f, $(f)_0 = \sum_{n_i \geq 0} n_i P_i$. The partial sum with $n_i < 0$ will be called the pole-divisor of f, $(f)_\infty = -\sum_{n_i \leq 0} n_i P_i$. The collection of all zeroes will be called the zero set of f, zero-set$(f) = \{P_i : n_i > 0\}$. The collection of all poles will be called the pole set of f, pole-set$(f) = \{P_i : n_i < 0\}$. The collection of all zeroes and poles will be called the support of f, supp$(f) = \{P_i : n_i \neq 0\}$. ■

Example 3: Let us consider P^1_R where R is the real field. Let $P = (x^2 + 1)$, P_∞ be the point at ∞ and $f = x^2 + 1$. Then $(f)_0 = P$ and $(f)_\infty = 2P_\infty$. While $\mu(P) = 2$. Therefore

$d((f)_0) = 2 = d((f)_\infty)$.

■

So far, we just defined some concepts which are formal. Since the divisors are formal sums of points, even if they form a group under formal summation, it seems that they can not help us in the discussions of curve theory. However some refined structures of them turn to be powerful tools. We shall start the journey with

Definition : Two curves C, C' are said to be *birationally equivalent* iff the rational function fields $F(C), F(C')$ of them are isomorphic.

■

The folloowing example shows that not nay two curves are birationally equivalent,

Example 4: Let K be an algebraically closed field field with ch(K)=p. Let us consider the following Fermat curves F_n,

$$x_1^n + x_2^n = x_0^n$$

where $n > 2$ and $p \dagger n$ (in case that $p = 0$, the second condition is void).

We claim that F_n is not *birationally equivalent* to P_K^1, or the rational function field $F(F_n) \neq K(t)$. Suppose it is, we may dehomogonize the defining equation by setting $x_0 = 1$, and let $x_1 = g(t)/f(t), x_2 = h(t)/f(t)$ with $(f(t), g(t), h(t)) = (1)$. We wish to deduce a contradiction if f, g, h are not all in the field K. Suppose that there are triples f, g, h, we will take a triple f, g, h with $max\ (deg(f), deg(g), deg(h))$ the smallest possible one. Note that if two of f, g, h are constants, then the third one must be constant which is impossible. The above defining equation becomes

$$g(t)^n + h(t)^n = \prod_{i=0}^{n-1} (g(t) + \omega^i h(t)) = f(t)^n$$

Since each prime factor $p(t)$ of two of $(g(t) + \omega^i h(t))$'s must be a prime factor of g, h and hence a prime factor of f, therefore all $(g(t) + \omega^i h(t))$ are co-prime. We may rewrite the above equation as

$$g(t) + \omega^i h(t) = c_i \alpha_i^n(t)$$

and $\sum\ deg(\alpha_i) = deg(f(t))$. Now we select three of them with one of α_i non-constant. Take the three corresponding equations, and eliminate g, h, we get a new equation of the following form

$$e_i \alpha_i^n(t) + e_j \alpha_j^n(t) + e_k \alpha_k^n(t) = 0$$

Absorbing the coefficients e_i, e_j, e_k to the polynomials, we rewrite the above as

$$\beta_i^n(t) + \beta_j^n(t) = \beta_k^n(t)$$

Note that $max\ (deg(f), deg(g), deg(h)) > max\ (deg(\alpha_i), deg(\alpha_j), deg(\alpha_k))$. A contradiction.

■

Exercises

(1) Let us assume that ch(K)>2. Let us consider a curve, a projective *elliptic curve* **C** in the plane, whose affine part is defined by $x_2^2 = (x_1 - 2)(x_1 - 1)(x_1 - 0)$. Show that **C** is not birationally equivalent to P_K^1.

(2): Let us consider P_K^1 with the ring of regular functions $K[x]$. Find the divisor (x).

(3): Let a curve C with its affine part defined by $x^2 + y^3 + x^3 = 0$. What is the equation which defines its projective completion?

(4): Let a projective curve C with its affine part defined by $x^2 + y^3 + x^3 = 0$. What is the divisor (x)?

(5): Find the intersection number $I(P, F \cap G)$ for $P = (9, 0)$, $F = x + y + x^3$, $G = x^2 - y^4 + x^3$.

4.6 Riemann's Theorem

An important mathematical approach is to study the relations between the geometric sets and the algebraic objects defined over the sets. One example is the *Poincare conjecture* : the homotopy groups of the sphere and the spheres determine each other. In fact, before Poincare, R. Descartes studied the relations between the functions which defined the algebraic curves. As we pointed out in the preceding subsection, any non-constant function will have some pole. i.e., if we consider only functions without poles, then we only have constants for any projective curve. The set of no-pole functions tells us nothing of the sets. .

We will relax the condition a little bit. Let us consider all functions with some assigned poles. Then can we tell the differences of the underlying projective curves? Let us consider the following example.

Example 1: Let us consider the simplest curve, a straight line in the plane, defined by $x_2 = 0$. Let us consider the pole set $\{x_0 = 0, x_1 = 1, x_2 = 0\} = \{P\}$. It is not hard to see that the set of all functions with at most one pole at P is $\{(a_0x_0 + a_1x_1)/x_0\}$ which is a vector space of dimension 2 over K. It is not hard to see that the set of all functions with at most nP as poles is $\{(a_0x_0^n + a_1x_0^{n-1}x_1 + \cdots + a_nx_1^n)/x_0^n\}$ which is **a vector space of dimension** $n+1$ over K. Let $t = x_1/x_0$. Then $a_0x_0^n + a_1x_0^{n-1}x_1 + \cdots + a_nx_1^n)/x_0^n$ can be re-written as $a_0 + a_1t + \cdots + a_nt^n$, a polynomial in t of degree $\leq n$. ■

Example 2: Assume that $ch(K) \neq 2$ and K contains a square root of any element in K. Let us consider another curve, an *elliptic curve* C in the plane, defined by $x_2^2x_0 = (x_1 - ax_0)(x_1 - x_0)(x_1 - 0)$, where $a \neq 1 \neq 0$. Let us consider the pole set $\{x_0 = 0 = x_1, x_2 = 1\} = \{P\}$ Let us consider the set of all functions $L(nP)$ with at most n poles at P where n is a non-negative integer and no pole elsewhere. We **claim** that $L(nP)$ is **a vector space of dimension n** over K.

We shall dehomogonize the equation by setting $x = \frac{x_1}{x_0}, y = \frac{x_2}{x_0}$. Then the defining equation can be rewritten as

$$g(x,y) = y^2 - (x-a)(x-1)(x-0)$$

In general, the functions $f(x,y)$ in $F(C) = K(x)[y]/g(x,y)$ are of the following form

$$f(x,y) = \frac{g_0(x) + g_1(x)y}{h(x)}$$

We may assume that $(h(x), g_0(x), g_1(x)) = (1)$, otherwisei we may reduce the form. We make a further assumption that $h(x)$ splits completely (otherwise go to a finite extension of K) Suppose that $f(x,y)$ has no pole at finite distancei (means in A_K^2). We **claim** that $h(x)$ is a non-zero constant.

If not, we will show that f has a pole at finite distance, thus $f \notin L(nP_\infty)$. Consider any non-constant factor $x-\beta$ of $h(x)$. We have two cases, (1) $\beta \neq 0, 1, a$, (2) $\beta = 0, 1, a$. In the case (1) the intersection of $x-\beta$ with curve C will be distinct points P_1 and P_2 (corresponding to two distinct non-zero values of y on C with $x = \beta$). Either $g_0(\beta) \neq 0$ or $g_1(\beta) \neq 0$ but not both, since $(h(\beta), g_0(\beta), g_1(\beta)) = 1$. In any situation, the numerator of $f(x,y)$ can not be 0 at both non-zero values of y. Therefore, f must have a pole at finite distance. In the case (2), the intersection of $x-\beta$ with the curve C is at the point $P = (\beta, 0)$ twice. If $g_0(\beta) \neq 0$, since $y = 0$, then the numerator does not pass the point $P = (\beta, 0)$, so the function will have two poles at $P = (\beta, 0)$. It is impossible. Therefore $g_0(\beta) = 0$. We shall study the completion of the local ring O_P at the point P. Let us discuss the situation that $\beta = 0$ (other situations are similar). The defining equation is

$$y^2 = (x-a)(x-1)(x-0)$$

It is easy to see y is a uniformization parameter and $\hat{O}_P = K[[y]]$. We have

$$\begin{aligned} x &= a^{-1}y^2 + \cdots \\ g_0 &= b_2 y^2 + \cdots \\ g_1 &= c_0 + c_1 y + \cdots \end{aligned} \tag{1}$$

where $c_0 \neq 0$. It is easy to conclude that the numerator of f has P as zero once. Therefore f has a pole at finite distance. Our claim is thus proved.

At P_∞, we set $u = \frac{x_0}{x_2} = y^{-1}, v = \frac{x_1}{x_2} = xy^{-1}$. The defining equation becomes

$$u = (v - au)(v-u)(v-0)$$

It is easy to see that $ord_{P_\infty}(v) = 1, ord_{P_\infty}(u) = 3$. Hence $ord_{P_\infty}(y) = -3$, and $ord_{P_\infty}(x) = -2$. Therefore, a polynomial of the form $g_0(x) + g_1(x)y$ will have order $-2i$ or $-2j-3$ at P_∞. It is easy to see that there is no function with a simple pole at P_∞ and all functions with at most n poles at P_∞ form **a vector space of dimension** n. ■

Apparently, by study $L(nP)$, we can tell the difference between two curves discussed as above. We will define,

Definition : Let $D = \sum_i n_i P_i$ be a divisor. If all $n_i \geq 0$, we say that D is *effective* , in symbol, $D \succ 0$. We define $L(D)$ as

$$L(D) = \{f \in F(X) : (f) + D \succ 0\}$$

and we define $\ell(D) = dim_K(L(D))$ ■

There are many relations between $\ell(\mathbf{D})$ and $d(\mathbf{D})$. For instance, if $D \succ 0$, it can be shown that $\ell(D) \leq d(D)+1$. Let us prove it for a simple case that $D = nP$ with P a rational point, i.e., $\hat{\mathrm{O}}_P = K[[t]]$. Let us define a map $\pi : L(D) \mapsto K^n$ as $\pi(f) = (a_1, \cdots, a_n)$ where $f \in L(D)$ and $f = a_n t^{-n} + \cdots + a_1 t^{-1} + \cdots$. Then clearly π is a linear map and ker(π) =K. Therefore $\ell(D) \leq n + 1 \leq d(D) + 1$.

Similarly, it is easy to see that if $\mathbf{E} \succ \mathbf{D}, i.e., E - D = H \succ 0$, then $dim(L(\mathbf{E})/L(\mathbf{D})) = \ell(\mathbf{E}) - \ell(\mathbf{D}) \leq d(\mathbf{D}) - d(\mathbf{E})$, i.e.

$$\ell(\mathbf{E}) + d(\mathbf{E}) \leq \ell(\mathbf{D}) + d(\mathbf{D})$$

Especially, if $\mathbf{D} = \mathbf{E} + P$ for some rational point P, then we have

$$\ell(\mathbf{E}) \leq \ell(\mathbf{D}) + 1$$

Proposition 4.6.1 *: We always have $\ell(\mathbf{D} + d(\mathbf{D}) \leq \ell(\mathbf{E}) + d(\mathbf{E})$ for any two divisors $\mathbf{E}$ and $\mathbf{D}$ with $\mathbf{E} \succ \mathbf{D}$*

Proof. Chevalley page 21 [8]. ■

We have the following Riemann's Theorem,

Proposition 4.6.2 *(Riemann's Theorem); Let X be a given projective curve and D any divisor. Then there is a minimal non-negative g, which will be called the* **genus** *of X, such that*

$$\ell(D) \geq d(D) + 1 - g$$

Moreover, if $d(\mathbf{D}) \geq 2g - 1$, than

$$\ell(D) = d(D) + 1 - g$$

Proof. Chevalley page 22 [8]. ■

Proposition 4.6.3 *: If $d(\mathbf{D}) < 0$, then $\ell(\mathbf{D}) = 0$.*

Proof. Let $f \in L(\mathbf{D})$. Then we have $(f) + \mathbf{D} \succ 0$ and $d(\mathbf{D}) = d((f)) + d(\mathbf{D}) \geq 0$. A contradiction. ■

We shall mention the following classical *Plücker's formula* without proof,

(ücker's formula: Let C be a non-singular plane curve of degree n. Then the genus g of C is given by the following formula,

$$g = \frac{(n-1)(n-2)}{2}$$

■

Example 3: Let us consider the *elliptic curve* of **Example 2**. It is easy to see it is non-singular. Then by Plücker's formula, its genus g is 1. It follows from Riemann's Theorem that $\ell(\mathbf{D}) \geq d(\mathbf{D}) = n$ for $\mathbf{D} = nP_\infty$ which has been verified by direct computation. ■

Example 4: Let us consider a Fermat's curve $x_1^3 + x_2^3 + x_0^3$ over F_{2^m}. Then by previous discussions, we know it is non-singular. By Plücker's formula, its genus g is 1. It follows from Riemann's Theorem that $\ell(D) \geq d(D)$. Let us verify Riemann's Theorem for some special divisors.

Let $P_\infty = (0,1,1)$. Let us consider $D = nP_\infty$ for some non-negative integer n. Note that $d(D) = n$. Let us make a projective transformation π

$$\begin{aligned}
(1) &: \pi(x_1) = y_1 + y_2 \\
(2) &: \pi(x_2) = y_2 \\
(j) &: \pi(x_0) = y_0
\end{aligned}$$

Then the defining equation becomes $y_1^3 + y_2 y_1^2 + y_2^2 y_1 + y_0^3$. Let us consider the affine part defined by $y_1 = 1$ with $x = y_0, y = y_2$. The equation becomes

$$y^2 + y = x^3 + 1$$

The function field $F(C)$ is of degree 2 over $\mathbf{F}_{2^m}(x)$. In general, the functions in $F(C) = K(x)[y]$ are of the following form

$$f(x,y) = \frac{g_0(x) + g_1(x)y}{h(x)}$$

We may assume that $(h(x), g_0(x), g_1(x)) = (1)$. We make a further assumption that $h(x)$ splits completely (otherwise go to a finite extension of K or use some algebra). Suppose that $f(x,y)$ has no pole at finite distance, we claim that $h(x)$ is a non-zero constant.

If not, we will show that f has a pole at finite distance, thus $f \notin L(nP_\infty)$. Consider any non-constant factor $x - \beta$ of $h(x)$. The intersection of $x - \beta$ with curve C will be distinct points $P_1 = (\beta, y_1)$ and $P_2 = (\beta, y_2)$ (corresponding to two distinct non-zero values, y_1, y_2, of y on C with $x = \beta$, which can always be achieved if we go to an algebraic extension of K). If the numerator of $f(x,y)$ is zero at both points, then we have

$$\begin{aligned}
g_0(\beta) + y_1 g_1(\beta) = 0 \\
g_0(\beta) + y_2 g_1(\beta) = 0
\end{aligned}$$

If we treat y_1, y_2 as numbers and $g_0(\beta), g_1(\beta)$ as variables, then we conclude

$$g_0(\beta) = 0$$
$$g_1(\beta) = 0$$

which contradicts our assumption that there is no common factor for $h(x), g_0(x), g_1(x)$! Our claim is thus proved.

At $P_\infty = (0,0,1)$, we set $u = \frac{y_0}{y_2} = xy^{-1}, v = \frac{y_1}{y_2} = y^{-1}$. The defining equation becomes

$$v^2 + v = u^3 + v^3$$

It is easy to see that $ord_{P_\infty}(v) = 3, ord_{P_\infty}(u) = 1$. Hence $ord_{P_\infty}(y) = -3$, and $ord_{P_\infty}(x) = -2$. Therefore, a polynomial of the form $g_0(x) + g_1(x)y$ will have order $-2i$ or $-2j-3$ at P_∞. It is easy to see all functions with at most n poles at P_∞ form a vector space of dimension n.

■

Example 5: Let us consider the curve $x^2 - y^2 + x^3 + y^4$ defined over the complex numbers **C**. By Jacobian criteria, the origin is the only singularity with ordinary double multiplicity. Hence by an **extended** Plücker formula which works for singular curves, its genus g is $\frac{(4-1)(4-2)}{2} - 1 = 2$ (cf. [13]).

■

From the above we easily conclude that non-singular plane curves of degrees 1,2,3,4,$\cdots$ are of genus 0,0,1,3,$\cdots$. If there is a plane curve of genus 2, then there must be some singular points in any planar model. If we consider all algebraic curves, the coarsest classification is by their *genus*, i.e., two curves are in the same class if their genus are the same. This is the *discrete* parameter of algebraic curves according to their (distinct) genuses. If the ground field is the complex number field **C**, then it simply states all algebraic curves with same given genus the underlying topological spaces are the same. However, *analytically*, the structure can still be different, thus we have a further fine parameter for all algebraic curves over the complex field C with the same genus, namely, the variety of moduli of curves of genus g, which is a point if $g = 0$, has dimension 1 if $g = 1$, and dimension $3g - 3$ if $g \geq 2$. If the ground field is not algebraically closed as **C**, then we have the following Proposition,

Proposition 4.6.4 *: Let X be an algebraic curve. Then $F(X) \equiv K(x) \Leftrightarrow X$ has a rational point and $g = 0$.*

Proof. Chevalley page 23 [8].

■

Example 6: Let C be a plane curve over $K = F_q(u,v)$, where u, v are symbols, defined by

$$ux^m + vy^m = 1$$

with $p \dagger m$. It follows from Plücker's formula that its genus $g = (m-1)(m-2)/2$. However there is no rational point. If there is one, let its coordinate be $(g(u,v)/f(u,v), h(u,v)/f(u,v))$, then the above equation can be rewritten as

$$ug(u,v)^m + vh(u,v)^m = f(u,v)^m$$

Let $deg(f(u, v)) = a, deg(g(u, v)) = b, deg(h(u, v)) = c$. Then the degrees of those three terms are $1 + mb, 1 + mc, ma$ respectively. Moreover the highest two of them must be equal. We conclude that the highest form of $ug^m(u, v)$ and $vh^m(u, v)$ must cancel out which is impossible.

Since there is no rational point, for $m = 2$, the curve C is with genus $g = 0$ while not a projective line.

■

Exercises

1 Show that the curve defined over F_{3^2} by the following equation

$$ax^2 + by^2 = 1$$

is a projective line for any $0 \neq a, b \in F_{3^2}$.

(2); Finish the arguments of **Example 5**.

(3): Finish the arguments of **Example 6**.

(4): Prove Riemann's Theorem for P^1_K directly.

(5): Given $ch(K) > 2$, and the curve C defined by $x^2 - y^2 + x^3 + y^4 = 0$. Find $L(P), L(2P), L(3P)$.

4.7 Riemann-Roch Theorem

Note that the Riemann's theorem is an inequality. There is a number missing for it to become an equality. Roch, a student of Riemann, found a natural explaination of the missing number. First, we have to introduce the concepts of *differentials* and *canonical divisors*. Classically, we have

(1) Let us consider the complex case, where $K = C$ the field of complex numbers. Let us assume that the reader is familiar with the concept of residue in *complex analysis of functions of one variable*. We have the following formula with $res_{P_j} f(x)d\ x$ the *residue of* $f(x)d\ x$ at a point P_j,

$$\frac{1}{2\pi i}\int_\Gamma f(x)dx = \sum res(f(x)dx)$$

where Γ is a simple loop such that $f(x)dx$ has no pole on the loop and the sum is over all interior points of the loop where $f(x)$ has a pole. If we take the loop small enough so there is no pole inside, then the integral is 0. Therefore, we take the same loop with the reverse direction, then the integration is still 0, which means $\sum res_{P_j}(f(x)dx) = -0 = 0$. That is an important classical theorem: $\sum res_{P_j}(f(x)dx) = 0$.

(2) The Chinese remainder theorem states that for $p_j^{m_j}$ where p_j are distinct prime numbers $j = 1, \cdots n$ and any set of integers $\{r_j\}$, there $jsanintegerrsuchthatr \equiv r_j\ mod\ (p_j^{m_j})$

We will get similar statements for the rational function field $K(C)$ of an algebraic curve C. For the statement (2), given $f_j \in K(C)$, we are required to find a rational function f such that

$$ord_{P_j}(f - f_j) \geq m_j$$

It is true and known as the *approximation theorem* (see below). In the above statement (1), we rely on a sense of *topology* or *continuity*. In our present situation, we are without them. However, the same theorem is true in any case due to **Approximation Theorem** of section 4.4. From another point of view, we may rewrite (1) as

$$\sum res(f(x)dx) = \frac{1}{2\pi i}\int_\Gamma f(x)dx =< f(x)dx \mid \Gamma >=< f(x)dx \mid \sum_{P_j \ is \ insides \ \Gamma} P_j >$$

we may think that $f(x)dx$ is in the *dual space* of $\sum P_j$, i.e., by the *residue* map, every differential assigns a value (i.e., the residue) to a point P_j. A. Weil found the correct algebraic way of treating the differentials. We will use the concept of A. Weil in defining *repartition* and *differential* algebraically for the general cases as follows (C. Chevalley [8]),

Definition : A *repartition* ξ is an assignment to every point P a function ξ_P such that there are only finitely many points P with $ord_P(\xi_P) < 0$. The collection of all *repartitions* is denoted by Ξ. It is clear that we may assign a function f to all points P_j. It is called the *constant repartitions.* Their collection of constant repartitions is isomorphic to $F(C)$ and will be denoted by the same symbol $F(C)$. Given any divisor D, we will define $\Xi(D)$ as the collection of all repartition ξ such that

$$ord_P(\xi_P) + ord_P(D) \geq 0$$

for all P.

Classically, we have that a differential ω vanishes on $F(C)$ and $\Xi(\mathbf{D})$ for some suitable divisor $\mathbf{D}$ (see below, Example 1). We give the following general definition,

Definition : A *differential* ω is a (K-)linear functional on Ξ such that there exists a divisor $\mathbf{D}$ with ω vanishs identically on $\Xi(\mathbf{D}) + F(C)$. Let P be a point and ω a differential. Let $1/P$ be a repartition whose value at P is 1 and 0 elsewhere. Then we define the *residue* of ω at P, $res_P(\omega)$, as

$$res_P(\omega) = \omega(1/P)$$

■

The above definitions are abstract and general. Moreover they provide a proof for the Riemann-Roch theorem (see below). The shortcoming is that they are too abstract to be used. In our present case of the fields K being algebraically closed or perfect field K, the above abstract definition means the following equivalent definitions which will be preferred and will be used in this book,

Definition : Let C be an absolutely irreducible and non-singular curve. Let $x_i, y_i \in F(C)$. A finite sum of the form $\sum_i x_i dy_i$ is called a *differential* of $F(C)$ where d is a derivation. The collection of all differentials is denoted by $\Omega_K(C)$. Let $\omega \in \Omega_K(C)$. Then the order of ω at a point P, $ord_P(\omega)$, is defined to be $ord_P(f)$ where $\omega = fdt$ and t is an uniformization parameter of $R[C]_P$ at P. We define $(\omega) = \sum ord_P(\omega)P$. We say $\omega \in \Omega(D)|Longleftrigghtarrow$ $() \succ D$. ■

To interpretate $\sum_i x_i dy_i$ as a differential (in the sense of Weil), we have to calculate the *residue* res_P at a point P. We have

Proposition 4.7.1 *Let P be a regular point with uniformization parameter t and $\sum x_i dy_i$ be a differential. Let $\sum x_i dy_i$ be expressed as*

$$\sum x_i dy_i = (\sum_r^{\infty} c_i t^i)dt$$

Then we have (1) $res_P(\sum f_i d(x)) = c_{-1}$, if the point P is rational. (2) In general,

$$res_P(\sum x_i dy_i) = Tr_{K'/K}(c_{-1})$$

Proof. Chevalley page 110 [8]. ■

Example 1: Let us consider the curve P^1_C where C is the field of the complex numbers. We wish to show that the *differentials* in the classical sense are equivalent to some special linear functions on Ξ. First, we will generalize our previous concepts as follows.

Let $f(x)dx$ be a differential in the classical sense. Let v_p = the order of $f(x)dx$ at point P, i.e., $f(x)dx = (\sum_{i=v_p}^{\infty} a_i t^i)dt$ where $a_{v_p} \neq 0$ and t is the uniformization parameter. Then $(f(x)dx) = \sum v_p P$. Let $D = \sum_i n_i P_i$ be a divisor. Then we have $f(x)dx \in \Omega(D)$ $\Longleftrightarrow (f(x)dx) \succ D$. We will separate the discussions in three steps: (1). Every differential in the classical sense induces a differential in the Weil's sense, and this map is an one to one correspondence. (2) and (3) the induced map is an bijection between $\Omega(D)$ and the linear functionals vanishing on $\Xi(D) + F(C)$.

(1): We will show that every *differential* in the classical sense can be viewed as a (K-)linear function on Ξ which vanishes on $\Xi(D) + F(C)$ for some suitable divisor D. Let $\omega = f(x)dx$ be a differential in the classical sense. Then for any *repartition* ξ and any point $P \in P^1_C$(note that all points are rational), let ξ_p be the function element specified by ξ at the point P, we may define

$$\omega_P(\xi) = the\ residue\ of\ \xi_p\omega\ at\ P$$

Since for the above residue to be non-zero at P, either ω must have a pole or ξ_P must have a pole, and both sets are finite, therefore there are only finitely many P for the above residue to be non-zeroes, we may define

$$\omega(\xi) = \sum_P \omega_P(\xi)$$

From the above we conclude that ω defines a linear function on Ξ. It is clear that for any constant repartition $\xi = g(x) \in F(C)$, $\omega(\xi) = \sum res(g(x)f(x)dx) = 0$.

For any differential $f(x)dx$ in the classical sense, let $(f(x)dx) = \sum_i n_i P_i - \sum_j m_j P_j$ where n_i, m_j are non-negative, let $D = \sum_i n_i P_i - \sum_j m_j P_j$. It is easy to see that $(f(x)dx) = D$,

$(f(x)dx) \succ D$ and $(f(x)dx) \in \Omega(D)$. Furthermore, $\omega_P(\xi) = 0$ for all $P, \xi \in \Xi(D)$. Therefore ω vanishes on $\Xi(D) + F(C)$. We define a map from the differentials in the classical sense $\in \Omega(D)$ to an element in the new sense which vanishes on $\Xi(D) + F(C)$.

Moreover, we claim that any non-zero differential $\omega = f(x)dx$ in the classical sense can not induce a zero linear function. Let a be any point that is neither a zero nor a pole of $f(x)$. Consider a *repartition* ξ defined by $\xi = (x-a)^{-1}$ at $P = (x-a)$ and 0 elsewhere. Then $\omega(\xi) = f(a) \neq 0$.

(2): We wish to show that the induced residue map is onto. We observe that the classical differential $\omega \in \Omega(D)$ will vanish on $\Xi(D) + F(C)$ for any divisor D. Therefore the induced map by the residue will send $\Omega(D)$ to a subspace of all linear functionals which vanish on $\Xi(D) + F(C)$. We wish to show that they have the same dimensions. Then clearly it will imply that the induced residue map is onto. We shall count the number of linear independent classical *differentials* in $\Omega(-D)$ for any divisor D. We claim that the number is $Max\{0, d(D) - 1\}$.

Let $-D = D_A + D_\infty$ where D_A is the affine part of $-D$ and D_∞ is the part of $-D$ at ∞. Consider $D_A = \sum n_i P_i$ with $P_i = (x - a_i)$ and $D_\infty = mP_\infty$. Let $h(x) = \prod_i (x-a_i)^{-n_i}$ and $f(x)dx \in \Omega(-D)$, i.e.,

$$(f(x)dx) \succ -D$$

Then $(f(x)dx) = \sum_i n_i' P_i + m' P_\infty + \sum_k \ell_k P_k$ with $n_i' \geq n_i$, $m' \geq m$ and $\ell_k \geq 0$. We see that $h(x)f(x)$ has no poles at finite distance, therefore it is a polynomial $g(x)$ in x. Therefore (g(x)dx) is $\sum (n_i' - n_i)P_i + (m' + (\sum_i n_i)P_\infty$. Let $t = x^{-1}$ be the uniformization parameter at ∞. We have $dx = -t^{-2}dt$. Then it follows that at ∞,

$$\begin{aligned} &v_\infty(h(x)f(x)dx) = m' + (\sum_i n_i) \geq m + (\sum_i n_i) = -d(D) \\ &v_\infty(g(x)) \geq 2 - d(D), \end{aligned}$$

in other words, $g(x)$ is a polynomial of degree $\leq d(D) - 2$ which (and hence f(x)) forms a vector space of dimension $d(D) - 1$ or 0.

(3): We claim that the number of linear independent linear functions on Ξ which vanish on $\Xi(-D) + F(C)$ is $Max\{0, d(D) - 1\}$.

Let us discuss a simple case, say $D = P \neq P_\infty$. Let ξ be any repartition. It is easy to see that there is a rational function $f(x)$ such that $\xi - f(x)$ has no-pole. Let $\xi_P(P) = a$ which may be 0. We shall consider $\xi - f(x) - a$, then this repartition has a zero at P, therefore it is in $\Xi(-P)$ and $\xi \in \Xi(-P) + F(C)$. We conclude that a linear function which vanishes on $\Xi(-P) + F(C)$ must vanish on any repartition ξ, which means the differential which vanishes on $\Xi(-P) + F(C)$ must be the zero linear function. Our claim is proved in this case. For the general cases, the reader is referred to Chevalley p.26-30 [8] or Exercise 4.

According to $(1), (2), (3)$, *differentials* in the classical sense are the linear functions which vanishing on $\Xi(D) + F(C)$.

■

Example 2: Let $C = P_K^1$ with $ch(K) > 2$. Then $F(C) = k(x)$. Let $\omega = (x^2+x^3)d(x^2)+dx$ be a differential. It is easy to see that $\omega = (1+2x^3+2x^4)dx$. At any point $P_a = ((x-a))$. the local ring at P_a is $R = K[x]_{(x-a)}$ with maximal ideal $(x-a)R$ where $(x-a)$ is the uniformization parameter. Then $\omega = (1+2(x-a+a)^3+2(x-a+a)^4)d(x-a) = f(x-a)d(x-a)$. We have $ord_{P_a}(\omega) = ord_{P_a}(f(x-a))$. We have one more point P_∞ at ∞. The local ring at P_∞ is $K[\tau]_{(\tau)}$ where $\tau = \frac{1}{x}$. The differential $\omega = (1+2\tau^{-3}+2\tau^{-4})d(\tau^{-1}) = -(\tau^{-2}+2\tau^{-5}+2\tau^{-6})d\tau) = -\tau^{-6}(2+2\tau+\tau^4)d\tau$. So we have that $ord_{P_\infty}(\omega) = ord_{P_\infty}(-(\tau^{-2}+2\tau^{-5}+2\tau^{-6}) = -6$.

■

Some of the important results are the following,

Proposition 4.7.2 *Let ω be a differential with poles P_i, then we have*

$$\sum res_{P_i}(\omega) = 0$$

Proof. Chevalley page 49 [8]. ■

We may view the following Proposition as the *independence of points*,

Proposition 4.7.3 *(The approximation theorem for functions) Let $P_1, \cdots, P_m$ be finitely many points. Let $f_1. \cdots, f_m$ be m functions and $n_1, \cdots, n_m$ m integers. Then there is a function f, such that*

$$ord_{P_i}(f - f_i) \geq n_i \ for\ all\ i$$

Proof. Chevalley page 11 [8]. ■

Proposition 4.7.4 *(The existence theorem for differentials) Let $P_1, \cdots, P_m$ be finitely many distinct rational points. Let $\mathbf{D} = \sum P_i$. Given $r_i \cdots, r_m$ be elements in K respectively, such that*

$$\sum_i r_i = 0$$

Then there exists a differential ω of F with the divisor $(\omega) \in \Omega(D)$ such that $res_{P_i}\omega = r_i$.

Proof. Chevalley page 50 [8]. ■

Remark: If there are more rational point P', then we may push the residue to P', i.e., define $r' = -\sum_i r_i$, then the condition $\sum_i r_i = 0$ is void. Note that $\Omega(D+P') \subset \Omega(D)$.

■

Example 3: The reason we require the point being rational can be illustrated by an example. Let us consider the curve $\mathbf{P^1_K}$ and the following differential

$$\eta = \frac{2xdx}{x^2+1}$$

Let the ground field K be $\mathbf{C}$ the compex field. By partial fraction, we have

$$\eta = (\frac{1}{x+i} + \frac{1}{x-i})dx$$

It is easy to see that the residues at $(x+i),(x-i)$ are $1,1$ respectively. At P_∞ with uniformization parameter t, η becomes

$$\eta = (-2t^{-1} + \cdots)dt$$

It is easy to see that $\sum res_{P_j}\eta = 0$. However, let us consider the case that the ground field $K = \mathbf{R}$ the real field. At ∞, it has a residue -2. There is only 1 point $P = (x^2+1)$ at finite distant with a pole. Let $t = x^2+1$ be an uniformization parameter at this point. Then we have

$$\eta = t^{-1}dt$$

The coefficient of t^{-1} term is 1. Its residue will be defined as the *trace* of 1, i.e.,

$$res_P(\eta) = Tr_{\mathbf{C}/\mathbf{R}}(1) = 2$$

where $Tr_{\mathbf{C}/\mathbf{R}}$ is the *trace* operation from $\mathbf{C}$ to $\mathbf{R}$. We still have $\sum res_{P_i}\eta = 0$.

Let us consider another differential ω defined as

$$\omega = \frac{2dx}{x^2+1}$$

Let the ground field K be $\mathbf{C}$ the compex field. By partial fraction, we have

$$\omega = (\frac{i}{x+i} + \frac{-i}{x-i})dx$$

It is easy to see that the residues at $(x+i),(x-i)$ are $i,-i$ respectively. At P_∞ with uniformization parameter t, ω becomes

$$\omega = (-2 + \cdots)dt$$

It is easy to see that $\sum res_{P_j}\omega = 0$. However, let us consider the case that the ground field $K = \mathbf{R}$ the real field. At ∞, it has a residue 0. There is only 1 point $P = (x^2+1)$ at finite distant with a pole. Let $t = x^2+1$ be an uniformization parameter at this point. Then we have (cf **Example 3** of the preceding section), where we show that $x = i\sqrt{1-t}$ in the complete local ring $\mathbf{C}((\mathrm{t}))$.)

$$\omega = (x(t^{-1}))dt = (it^{-1} + \cdots)dt$$

The coefficient of t^{-1} term is i. Its residue will be defined as the *trace* of i, i.e.,

$$res_P(\omega) = Tr_{\mathbf{C}/\mathbf{R}}(i) = 0$$

where $Tr_{\mathbf{C}/\mathbf{R}}$ is the *trace* operation from $\mathbf{C}$ to $\mathbf{R}$. We still have $\sum res_{P_i}\eta = 0$. ■

Example 4: Let us consider the algebraic curve P_K^1. Let $\eta = f(x)dx = \frac{r(x)}{s(x)}dx$ be any non-zero differential. For simplicity, we assume that $s(x)$ can be factored into linear polynomials (otherwise go to a finite field extension of K). By the theory of partial fractions, we have

$$\eta = (u(x) + \sum_{i,j,a_i} \frac{c_{ij}}{(x-a_i)^j})dx$$

The sum of all residues at finite distance is clearly $\sum c_{i1}$. We shall show that the residue at P_∞ is the negative of it. Since the residue map is linear, it suffices to check every terms in the above formula. Note that at P_∞, a uniformizing parameter is $x^{-1} = t$. Then we have

$$\begin{aligned}
&u(x)dx = -t^{-2}u(t^{-1})dt \\
&(\frac{c_{ij}}{(x-a_i)^j})dx = -t^{-2}(\frac{c_{ij}t^j}{(1-a_it)^j})dt \ if \ j \geq 2 \\
&(\frac{c_{i1}}{(x-a_i)})dx = -t^{-2}(\frac{c_{i1}t}{(1-a_it)})dt
\end{aligned}$$

Clearly the residue at P_∞ is $-\sum c_{i1}$ and we have $\sum res_{p_i}(\eta) = 0$. ■

Definition : Let $\sum f_i(x)dx$ be a differential of a projective non-singular curve C. Then the divisor, $(\sum f_i(x)dx)$, of $\sum f_i(x)dx$ is defined to be $\sum_P (ord_P(\sum_i f_i(x)dx))P$ which is called a *canonical divisor* of C. ■

Exercises

(1): Given the curve $C = P_K^1$. Find a differential $f(x)dx$ with residues 1 at $x = 1$ and -1 at $x = -1$ with 0 residues elsewhere.

(2): Prove the *existence theorem for differential* for P_K^1 directly.

(3): Prove the *approximation theorem for functions* for P_K^1 directly.

(4): Given the curve P_C^1 where C is the field of complex numbers, and any divisor $\mathbf{D}$, show that the dimension of all linear functional on Ξ which vanishes on $\Xi(-D) + F(C)$ is Max(0, d(D)-1).

(5): Show that in the ring of integers $\mathbf{Z}$, Proposition 4.7.2 is equivalent to the Chinese Remainder Theorem.

Proposition 4.7.5 *: Let W be a canonical divisor. Then we have (1) All canonical divisors W are linearly equivalent. (2) d(W)=2g-2. (3) $\ell(W)$=g where g is the genus of the curve C.*

Proof. (1) Easy. (2) Chevalley page 32 [8]. (3) Chevalley page 30 [8]. ■

Example 5: Let us consider **Example 2** of the preceding §. We have an equation,

$$y^2 = (x-a)(x-1)(x-0)$$

Let us consider the differentials dx, dy. By implicit differentiation, we have

$$dy = (\frac{3x^2 - 2(1+a)x + a}{2y})dx$$

Therefore dx and dy are linearly equivalent. Let us consider the differential dx. We will consider the curve C as a *covering* of the affine line $\mathbf{A^1_K}$. At finite distance of the affine line $\mathbf{A^1_K}$, let us consider (1) a rational point $(x-\beta)$ with $\beta = 0, 1, a$. (2) a rational point with $\beta \neq 0, 1, a$. (3) non-rational point $f(x)$ where $f(x)$ is a monic irreducible polynomial of degree > 1.

(1): Our defining equation provides $y = 0$ and y is an uniformization parameter. Then we have $x = y^2 + \cdots$ and $dx = (2y + \cdots)dy$. Therefore dx has a 0 of order 1 at these three points.

(2): The value of y must be $\pm\sqrt{(\beta - a)(\beta - 1)(\beta)} = \pm c$ where c may or may not in K. We will separate the discussions into two subcases.

(A): Suppose that $0 \neq c \in K$. Then $x - \beta$ is an uniformization parameter. Since $dx = d(x - \beta)$, then there is neither a zero nor a pole.

(B): Suppose that $c \notin K$. Then $x-\beta$ is an uniformization parameter. Since $dx = d(x-\beta)$, then there is neither zero nor pole.

(3): Note that with our basic assumption that the field K is perfect, $f(x), f'(x)$ have no common root. Clearly we have $df(x) = f'(x)dx$ or $dx = (\frac{1}{f'(x)})df(x)$. It can be shown that $f(x)$ is a local uniformization parameter in $K[x]_{(f(x))}$ and at the corresponding point(s) on the curve C. Since the residue of $(\frac{1}{f'(x)})$ is not zero modulo $(f(x))$, then there is neither a zero nor a pole.

Let us consider the point at ∞. As pointed out in **Example 2** of the preceding §, $x = t^{-2}$ for an uniformization parameter t. Therefore $dx = -2t^{-3}dt$, i.e., it has three poles at ∞. We conclude that $(dx) = P_0 + P_1 + P_a - 3P_\infty$ and $d((dx)) = 0 = 2g-2$ and $g = 1$. ■

Definition : Let $\mathbf{D}$ be a divisor on $\mathbf{C}$ defined over K. Let $\Omega(\mathbf{D})$ denote the K-vector space of all rational differentials η on $\mathbf{C}$, defined over K, with divisor $(\eta) \succ \mathbf{D}$, together with the zero differential. Let the number i($\mathbf{D}$)=$\dim_K \Omega(\mathbf{D})$ be called the *index* of D. ■

Proposition 4.7.6 : *We always have*

$$i(D) = \ell(W - D)$$

where W is a canonical divisor.

Proof. Let $W = (fdx)$. We have

$$\begin{aligned} & g \in L(W - D) \Leftrightarrow \\ & (g) + (W - D) \succ 0 \Leftrightarrow \\ & (g) \succ -W + D \Leftrightarrow \\ & (g) + (fdx) \succ D \Leftrightarrow \\ & gfdx \in \Omega(D) \end{aligned}$$

■

Example 6: Let us use the notations of **Example 5**. Let $D = P_0 + P_1 + P_a - 3P_\infty$. Certainly, we know that $(dx) \succ D$. Let fdx be another differential such that $(fdx) \succ D$. Then clearly we have $(f) \succ 0$. Therefore $f = const$. Hence $i(D) = 1$. Let us consider another example. Let $D' = nP_\infty$ where n is a positive integer. Note that $d(W - D') < 0$. Therefore $0 = \ell(W - D) = i(D)$. ■

Let **C** be a nonsingular, absolutely irreducible, projective curve of genus g defined over the finite field K. Assume that **C** has rational points. We have the following Riemann-Roch Theorem:

Proposition 4.7.7 *(Riemann-Roch Theorem):Let us assume the preceding paragraph. Let W be a canonical divisor. We have*

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

Proof. Chevalley page 30 [8]. ■

Corollary 4.7.8 *: If $d(\mathbf{D}) > 2g - 2$, then $i(\mathbf{D}) = 0$.*

Proof. Clearly, $d(W - D) < 0$, and $0 = \ell(W - D) = i(D)$. ■

In the previous discussions, sometimes we extend the ground field from K to a finite extension K'. We have

Proposition 4.7.9 *: If K' is a finite separable extension of K (which is always true for K a perfect field, and K' a finite extension of K), then there is no change of genus g for the ground field extension.*

Proof. Chevalley page 99.[8]. ■

Example 7: Let us use the notations of **Example 5**. According to the Riemann-Roch Theorem, we have

$$\ell(D) - i(D) = 0$$

Since $i(D) = 1$, so $\ell(D) = 1$. A generator of $L(D)$ is $\frac{1}{y}$. Let us consider $D' = nP_\infty$. Then it follows from the Riemann-Roch Theorem that $\ell(D') = n$, since we have $i(D') = 0$ and $g = 1$. Clearly, the set $\{x^i y^j : j = 0, 1, \text{and} 2i + 3j \leq n\}$ forms a basis for $L(D)$. ■

Exercises

(1): Given a non-singular curve C. The Weierstrass gaps i is the integers i such that there is no rational function z which has pole only at P and $ord_P(z) = -i$. Show that there are precisely g Weierstrass gaps where g is the genus of C.

(2): For **Example 5**, show that $\ell(\mathbf{D}) = 1$ directly.
(3): Let P be a rational point on P^1_K. Show that $i(P) = 0$ and $\ell(P) = 2$.

4.8 Weil Conjecture and Hasse-Weil Inequality

Given an algebraic curve C, in the *algebraic geometry coding theory* it is important to know the number of rational points on the curve over a finite field F_q. Let us extend the ground field $\mathbf{F}_q$ to $\mathbf{F}_{q^r}$. Let N_r be the number of rational points of $\bar{X}$ over $\mathbf{F}_{q^r}$. This number matters in the theory of the *geometric Goppa codes.* The **Weil's conjecture** is a conjecture about the *zeta function* of $\mathbf{X}$ which is related to N_r;

Definition : The *zeta function* $Z(t)$ is defined as

$$Z(t) = exp(\sum_{r=1}^{\infty} N_r \frac{t^r}{r})$$

■

Proposition 4.8.1 *(Weil's conjecture for curves): We have for a non-singular projective curve of genus g,*

$$Z(t) = \frac{P_1(t)}{(1-t)(1-qt)}$$

where

$$P_1(t) = \prod_{i=1}^{2g}(1-\alpha_i t)$$

with $\mid \alpha_i \mid = \sqrt{q}$

Proof. The conjecture was proved by Weil, see [37]. ■

Remark: The Weil's conjecture for higher dimensional projective algebraic varieties is established by Deligne (1974). ■

Example 1: Let us consider $C = P^1_{q^r}$. It is easy to see that C has q^r rational points on $A^1_{q^r}$ and one point extra (the point at ∞). So there are q^r+1 points and $N_r = q^r+1$. Substitute it to the definition of $Z(t)$, we have

$$Z(t) = exp(\sum_1^{\infty}(q^r+1)(\frac{t^r}{r}) = exp(\sum_1^{\infty}(q^r)(\frac{t^r}{r})exp(\sum_1^{\infty}(1)(\frac{t^r}{r})$$

Recall that

$$ln(\frac{1}{1-x}) = -\sum_1^{\infty}(\frac{x^r}{r})$$

Therefore we deduce that

$$Z(t) = \frac{1}{(1-t)(1-qt)}$$

■

Proposition 4.8.2 *(Hasse-Weil's inequality): Let C be a non-singular, absolute irreducible projective curve. Then we have*

$$| N_1 - q - 1 | \leq 2g\sqrt{q}$$

Proof. We may deduce a proof of curves case from Hartshorne's book "Algebraic Geometry" V Ex 1.10, Appendix C Ex 5.7. ■

Example 2: Let $p \geq 5$ and $q = p^m$. Then the Fermat curve $x_0^{q-1} + x_1^{q-1} + x_2^{q-1}$ has no solution as projective curve. Therefore, $N_1 = 0$, and $g = \frac{(q-2)(q-3)}{2}$. It is easy to see Hasse-Weil's inequality is satisfied. ■

Example 3: Let us consider the curve C defined by $x_0^3 + x_1^3 + x_2^3$ over $\mathbf{F}_q$. Let us take $q = 2$. Let us count the number of rational points on C. Since $x_i^3 = x_i$. It is easy to see that two of x_i's must be 1 and the third is 0. So we have 3 rational points.

Let $q = 2^2$. Let us count the number of rational points on C. Note that $\alpha^3 = 1$ for any non-zero $\alpha \in \mathbf{F}_{2^2}$. It is impossible to have two of x_0, x_1, x_2 to be 0, Since then the third one must be 0 and $(0,0,0)$ is not a projective point. It is also impossible to have x_0, x_1, x_2 all non-zero. Note that then $x_i^3 = 1$ for all i, we will have the impossible situation of $1+1+1 \neq 0$. Therefore we must have exactly one zero. Note that if $x_i \neq 0$, then we may assume that $x_i = 1$, Then we have

(1): $x_0 = 0, x_1 = 1$, x_2 arbitrarily non-zero $\in \mathbf{F}_{2^2}$. There are 3 points.
(2): $x_1 = 0, x_0 = 1$, x_2 arbitrarily non-zero $\in \mathbf{F}_{2^2}$. There are 3 points.
(3): $x_2 = 0, x_1 = 1$, x_0 arbitrarily non-zero $\in \mathbf{F}_{2^2}$. There are 3 points.

So there are totally 9 points. It makes the non-restrictive inequality in Hasse-Weil's inequality an equality.

Let $q = 2^3$. Let us count the number of rational points on C. It is impossible to have two of x_0, x_1, x_2 to be 0, since then the third one must be 0 and $(0,0,0)$ is not a projective point. So we have two cases, (1) two of them are non-zero and the third zero. (2) all of them are non-zero.

Case 1. Let us consider $x_2 = 0$, $x_1 = 1$. Then we have

$$x_0^3 = 1$$

The above equation is satisfied only if $x_0 = 1$. Therefore, by taking $x_0 = 0$ or $x_1 = 0$, we conclude that there are 3 rational points.

Case 2. We may take $x_2 = 1$. Let α be a generator of the multiplicative group $F_{2^3}^*$. Then $x^7 = 1$ for any non-zero x. If $x_1^3 = 1$, then $x_1 = 1$ and we must have $x_0 = 0$. It is impossible. Therefore we have that $x_1 \neq 1$, or $x_1 = \alpha, \alpha^2, \cdots, \alpha^6$. We want to show that for every possible value of x_1, we may find one and only one values for x_0 to satisfy the equation. Let us consider $x_1 = \alpha^j$ with $1 \leq j \leq 6$. iLet x^0 $\bar{x_0}$ be two solutions with fixed x_1. Then we have

$$1 + \alpha^{3j} = \alpha^k = \alpha^{k+7} = \alpha^{k+14} = x_0^3 = \bar{x}_0^3$$

Then it is easy to see that

$$(\frac{\bar{x}_0}{x_0})^3 = 1$$

Therefore $\bar{x}_0 = x_0$, so the solution must be unique, Moreover, 3 is a factor of one of $k, k+7, k+14$,therefore the equation can be solved.

So there are totally 9 rational points.

Let $q = 2^4$. Let α, β be defined by the following equations

$$\alpha^2 + \alpha + 1 = 0$$
$$\beta^2 + \beta + \alpha = 0$$

It is to see that $\mathbf{F}_{2^2} = \mathbf{F}_2[\alpha]$ and $\mathbf{F}_{2^4} = \mathbf{F}_{2^2}[\beta]$. Let us count the number of rational points on C. It is impossible to have two of x_0, x_1, x_2 to be 0, since then the third one must be 0 and $(0, 0, 0)$ is not a projective point. So we have two cases, (1) two of them are non-zero and the third zero. (2) all of them are non-zero.

Case 1. Let us consider $x_2 = 0$, $x_1 = 1$. Then we have

$$x_0^3 = 1$$

The above equation is satisfied only if $x_0 \in \mathbf{F}_{2^2} \subset \mathbf{F}_{2^4}$. There are 3 possible points. Therefore, by taking $x_0 = 0$ or $x_1 = 0$, we conclude that there are 9 rational points.

Case 2. We may take $x_2 = 1$. Then the defining equation can be rewritten as

$$x_0^3 + 1 = x_1^3$$

Let $y = x_0^3 \neq 0$. We have

$$y^5 = 1$$
$$(y+1)^5 = 1$$

It is easy to see that the above equations have no common solutions. Therefore there is no rational point in this case. Totally we have 9 rational points.

We may use the *Weil Conjecture* for the same purppose of determine the numbers of points for all F_{2^r} once we know the number $N_1 = 3$ and $N_2 = 9$. Let us compute N_3. Note that

$$Z(t) = exp(\sum_{r=1}^{\infty} N_r \frac{t^r}{r})$$

$$Z(t) = \frac{P_1(t)}{(1-t)(1-qt)}$$

where

$$P_1(t) = \prod_{i=1}^{2g}(1-\alpha_i t) = (1-\alpha t)(1-\beta t)$$

since $g = 1$. Then we have

$$ln\ Z(t) = 3t + 9t^2/2 + N_3t^3/3 + \cdots = ln\ (1-\alpha t) + ln\ (1-\beta t) - ln\ (1-t) - ln\ (1-2t)$$
$$= (1+2+\alpha+\beta)t + (1+2^2+\alpha^2+\beta^2)/2t^2 + (1+2^3+\alpha^3+\beta^3)/3t^3 + \cdots$$

Compare coefficients of the powers of t, we have the following equations,

$$(1) 1+2+\alpha+\beta = 3$$
$$(2) 1+4-\alpha^2-\beta^2 = 9$$
$$(3) 1+2^3-\alpha^3+\beta^2 = N_3$$

Solve the first two equations, we have

$$\alpha = \sqrt{-2}, \beta = -\sqrt{-2}$$

Substitute into the equation (3), we conclude that $N_3 = 9$. It checkes with our previous computation.

Note that *Hasse* proved the following formula for any genus 1 non-singular projective curves,

$$N_r = 1 + q^r - \alpha^r - \beta^r$$

Our result check with the formula. ■

i**Example 4:** We want to show that the *zeta function* is not determined by the genus of the curve. Let us consider the following genus 1 curve,

$$y^2 = x(x-1)(x-\alpha)$$

over F_{2^2} with $\alpha \neq 0, 1$ in F_{2^2}. Then F_{2^2} consists of $0, 1, \alpha, \alpha+1$ and α satisfies the following equation,

$$\alpha^2 + \alpha + 1 = 0$$

Let $x = 0, 1, \alpha$, then $y = 0$. Let $x = \alpha + 1$, then $y = \alpha$, we have another point at ∞, therefore there are 5 rational points. In the previous example there are 9 points over F_{2^2}. Therefore the *zeta function* must be different. ■

Example 5: Let us consider curve C defined by $x_0^5 + x_1^5 + x_2^5 = 0$ over $\mathbf{F}_{2^4}$. Let us count the number of rational points on C. It is impossible to have two of x_0, x_1, x_2 to be 0, since then the third one must be 0 and $(0, 0, 0)$ is not a projective point. Let us consider the case that there is exactly one zero. Then we have

(1): $x_0 = 0, x_1 = 1,\ x_2^5 = 1$. Note that $x^5 + 1 \mid x^{15} + 1$ by the field equation for $\mathbf{F}_{2^4}$, therefore there are 5 distinct x_2 satisfying $x_2^5 = 1$. There are 5 points.

(2): $x_1 = 0, x_0 = 1$, similarly there are 5 points.

(3): $x_2 = 0, x_1 = 1$, similarly there are 5 points.

Suppose that there is no zero among x_0, x_1, x_2. We may take $x_0 = 1$. Let $x_1^5 = a \neq 0$ then $x_2^5 = 1 + a \neq 0$. In other words, $a \neq 0, 1$. Since $(x_1^5)^3 = 1 = (x_2^5)^3$, we must have $a^3 = (a+1)^3 = 1$. Or $a, a+1 \in \mathbf{F}_{2^2}$, Let $\mathbf{F}_{2^2} = \mathbf{F}_2[\alpha]$. Then $(a, a+1) = (\alpha, \alpha+1)$ or $(\alpha+1, \alpha)$. For the equations $x_1^5 = a$, $x_2^5 = 1 + a$, there are 5 distinct solutions for each, so there are 25 solutions for one pair $(a, a+1)$. There are 50 solutions. The total number of solutions is 65. Note that $g = 6$ and $\mid 65 - 6 - 1 \mid \leq 2 \times 6 \times 8$. The Hasse-Weil's inequality is satisfied.

■

Example 6: Let us consider the Klein quartic curve over $\mathbf{F_2}$(or $\mathbf{F_{2^2}}$) or $\mathbf{F_{2^4}}$ defined by the equation $x^3y + y^3 + x = 0$. Its genus is $g = 3$.

Let us count the number of rational points. It is clear that $x = 0 \Leftrightarrow y = 0$. We denote this point $(0,0)$ (or $(0,0,1)$ as the projective point) by P. It is easy to see that $(1,1)$ is not a point. After we homogonize the equation, it becomes $x^3y + y^3z + xz^3 = 0$. Let $z = 0$. Then the curve has other two points at ∞, $(1,0,0) = P_1$ and $(0,1,0) = P_2$ over any field K. So there are 3 rational points over $\mathbf{F_2}$.

$$P = (0,0), \qquad P_1 = (1,0,0), \qquad P_2 = (0,1,0)$$

Over $\mathbf{F_{2^2}}$, we have $x^3 = 1$ for all $x \neq 0$. Therefore the affine equation gets reduced to $y + 1 + x = 0$ if $x, y \neq 0$, i.e., $y = 1 + x$ where $x \neq 1, 0$. Let α be a field generator of $\mathbf{F_{2^2}}$ over $\mathbf{F_2}$, i.e., $\alpha^2 + \alpha + 1 = 0$. Then $(\alpha, \alpha+1) = P_3, (\alpha+1, \alpha) = P_4$ are the extra rational points on the curve. Therefore, over $\mathbf{F_{2^2}}$, there are 5 rational points. The two extra points are

$$P_3 = (\alpha, \alpha+1), \qquad P_4 = (\alpha+1, \alpha)$$

Let us consider the field $\mathbf{F_{2^4}}$. Let $\mathbf{F_{2^4}} = \mathbf{F_{2^2}}[\beta]$ with β satisfying $\beta^2 + \beta + \alpha = 0$. First, we consider $x = \alpha$. Note that $\alpha^3 = 1$. Therefore, the equation will be reduced to $y^3 + y + \alpha = 0$. It is easy to see that $y^3 + y + \alpha = (y + \alpha + 1)(y^2 + (\alpha+1)y + (\alpha+1))$. We have to solve $y^2 + (\alpha+1)y + (\alpha+1) = 0$. The two roots are $y = \alpha^2(\beta+1), \alpha^2\beta$. We name $(\alpha, \alpha^2(\beta)) = P_5$, $(\alpha, \alpha^2(\beta+1)) = P_6$. Similarly, we find $P_7 = (\alpha+1, (\alpha+1)^2(\beta+\alpha)$, $P_8 = (\alpha+1, (\alpha+1)^2(\beta+\alpha+1))$. Thus we have four more points,

$$P_5 = (\alpha, \alpha\beta+\beta), \quad P_6 = (\alpha, \alpha\beta+\beta+\alpha+1), \quad P_7 = (\alpha+1, \alpha\beta++\alpha+1), \quad P_8 = (\alpha+1, \alpha\beta+\alpha)$$

We may consider the possibility that $y = \alpha, \alpha+1$. Let $y = \alpha$. Then the equation will reduced to $\alpha x^3 + x + 1 = 0$. Replacing x by αu, and the equation becomes $u^3 + u + \alpha + 1 = 0$, then the equation is similar to the one we just have discussed. We conclude that there are four more points

$$P_9 = (\alpha\beta + \beta + 1, \alpha), \quad P_{10} = (\alpha\beta + \beta + \alpha, \alpha), \quad P_{11} = (\alpha\beta, \alpha+1), \quad P_{12} = (\alpha\beta + \alpha, \alpha+1).$$

Let us consider $y = \alpha\beta$. Then $P_{13} = (\beta, \alpha\beta)$. Let us consider $y = \alpha(\beta+1)$. Then $P_{14} = (\beta+1, \alpha\beta+\alpha)$. Let us consider $y = \alpha\beta + \beta + 1$. Then $P_{15} = (\alpha+\beta, \alpha\beta+\beta+1)$. Let us consider $y = \alpha\beta + \beta + \alpha$. Then $P_{16} = (\alpha+\beta+1, \alpha\beta+\beta+\alpha)$.

There are 17 rational points over $\mathbf{F}_{2^4}$. ■

Exercises

(1): Do not use the Propositions of this section. Show that a curve of genus 0 has at most $q+1$ rational non-singular points.

(2): Let $\{C_s\}$ be a sequence of curves with genus g_s and the number of rational non-singular point N_s. Show that if $N_s \mapsto \infty$, then $g_s \mapsto \infty$.

(3): Show that the numbers of rational points for the curve in **Example 3** satisfy the Hasse-Weil's inequality.

(4): Use **Example 3** and Weil's conjecture to compute the Weil's *zeta function* for that particular curve.

(5): What is the number of rational points for the curve in **Example 3** over the field $\mathbf{F}_{2^5}$?

(6): What is the number of rational points for the curve in **Example 3** over the field $\mathbf{F}_{2^6}$?

(7) Find the *zeta function* of the curve in **Example 4**.

PART IV

GEOMETRIC GOPPA CODES

Chapter 5

Algebraic Curve Codes

In 1981, V.D. Goppa discovered an amazing connection between the theory of algebraic curves over a finite field $\mathbf{F}_q$ and the theory of error-correcting block q-ary codes. The idea is quite simple and generalizes the well known construction of Reed-Solomon codes. The latter use polynomials in one variable over $\mathbf{F}_q$ (the rational functions over $P^1_{\mathbf{F}_q}$ with only a pole at ∞) and Goppa generalized this idea using rational functions on an algebraic curve. In coding theory, take any basis of the k-dimensional subspace code space, we may specify a basis $\phi_i\}$, and a map $\pi([a_1, \cdots, a_k]) = \sum_i a_i \phi.$, and thus define a coding. This part is consider to be trivial and besides our discussions. From now on we omit the coding parts. We shall concentrate on the code spaces.

Definition : (*algebraic geometric code* or *a geometric Goppa code* in *function form* $C_L(B, D)$). Let $\mathbf{C}$ be an absolutely irreducible smooth projective algebraic curve of genus g over $\mathbf{F}_q$. Consider an (ordered) set $B = \{P_1, P_2, \cdots, P_n\}$ of distinct $\mathbf{F}_q$ rational points on $\mathbf{C}$ and a $\mathbf{F}_q$-divisor $\mathbf{D}$ on $\mathbf{C}$. For simplicity let us assume that the support of $\mathbf{D}$ is disjoint from $\mathbf{B}$, therefore $f(P_i) \neq \infty$ for $f \in L(D)$, and all i. The linear space L($\mathbf{D}$) of rational functions on $\mathbf{C}$ associated to $\mathbf{D}$ yields the linear evaluation map

$$\begin{aligned} &Ev_B : L(\mathbf{D}) \mapsto \mathbf{F}_q^n \\ &f \mapsto (f(P_1), \cdots, f(P_n)) \end{aligned}$$

■

Remark 1: If we take $\mathbf{D}$ as aP for some point P, then it is called a *one point code.* ■

It is easy to see that the word space is $\mathbf{F}_{q^n}$ a n-dimensional vector space. What is the message space? We wish to find the parameters $[n, k, d]$ for this code. Clearly n=dim($\mathbf{F}_q^n$). A function $f \in L(D)$ is sent to $[0 \cdots 0] \Leftrightarrow f(P_i) = 0$ for all i, i.e., $f \in$ L(-$\mathbf{B}$) $\Leftrightarrow$ (since the supports of $\mathbf{B}$ and $\mathbf{D}$ are disjoint) $f \in L(\mathbf{D} - \mathbf{B})$. Therefore we have

$$k = \ell(\mathbf{D}) - \ell(\mathbf{D} - \mathbf{B})$$

Proposition 5.0.3 *: If $0 < d(\mathbf{D}) < n$, then*

$$k = \ell(\mathbf{D}) \geq d(\mathbf{D}) - g + 1$$

The minimal distance d satisfies

$$d \geq n - d(\mathbf{D})$$

We have the following

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

Proof. In this case we have $d(\mathbf{D} - \mathbf{B}) < 0$ and $\ell(\mathbf{D} - \mathbf{B}) = 0$. In view of Riemann's theorem, we have

$$k = \ell(\mathbf{D}) \geq d(\mathbf{D}) - g + 1$$

We will prove the inequality for d. Let $\mathbf{D}$ be written as the difference of two disjoint effective divisors $\mathbf{D} = \mathbf{D_0} - \mathbf{D}_\infty$. Let $0 \neq f \in L(\mathbf{D})$. Let $\mathbf{B'}$ be $\sum P_j$ where $f(P_j) = 0$. clearly we have $\mathbf{B} \succ \mathbf{B'}$. Note that if $d(\mathbf{B'}) \leq d(\mathbf{D})$, then $d(f, 0) = n - d(\mathbf{B'}) \geq n - d(\mathbf{D})$ which is the inequality we wish to prove.

The relation $(f)_0 - (f)_\infty + \mathbf{D}_0 - \mathbf{D}_\infty \succ 0$ means that $(f)_0 \succ \mathbf{D}_\infty$ and $(f)_0 \succ \mathbf{B'}$. Since $\mathbf{B}$ and $\mathbf{D}$ are disjoint, $\mathbf{B'}$, $\mathbf{D}_\infty$ are disjoint, then it is easy to conclude that $(f)_0 \succ \mathbf{D}_\infty + \mathbf{B'}$. Suppose that $d(\mathbf{B'}) > d(\mathbf{D})$. Let $d(\mathbf{D}) = d(\mathbf{D_0}) - d(\mathbf{D}_\infty) = r - s$. Then $d((f)_0) > s + d(D) = s + r - s = r \geq d((f)_\infty)$. Therefore $f = 0$. Contradict to our assumption that $f \neq 0$. Hence, $d = min\{d(v, 0) : v \neq 0\} = n - max\{numbers\ of\ zero\ coordinates\ of\ non - zero\ vectors\}$. We prove that

$$d \geq n - d(\mathbf{D})$$

For the last inequalities, the first one is the classical Singleton bound (Proposition 1.6.2). The second one comes from the sum of the inequalities for k and d. ■

Remark 2: The lower bounds for the rank and the minimal distance are called the **designed rank** and **designed minimal distance**. ■

Example 1: Let the curve $\mathbf{C}$ be the projective line $P^1_{F_q}$ over a finite field F_q. Let $n = q - 1$ and β be a generator of the cyclic group F^*_q. Let $B = (\beta, \beta^2, \cdots .\beta^n)$, and $D = (k - 1)P_\infty$ where P_∞ is the point at infinity and we assume that $k < n$. Then we have

$$L(D) = \text{the set of all polynomials of degree less than k over } F_q$$

It is then easy to see that this is the classical **Reed-Solomon code**. ■

Example 2: Let us consider curve C defined by $x_0^3 + x_1^3 + x_2^3$ over $\mathbf{F}_{2^2}$. It is a curve of genus 1. There are nine points. We may take eight of them and call them $P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8$, and call the other one P. Let $\mathbf{D} = mP$. Then it follow from the preceding Proposition that

$$k \geq m$$
$$d \geq 8 - m$$

If we select $m = 3$, then $k \geq 3, d \geq 5$ and the code will have at least a rate of information $\frac{3}{8}$ and correct at least 2 errors.

■

Originally, Goppa used the dual construction using differentials on **C**. as follows,

Definition :(**algebraic geometric code** or **a geometric Goppa code** in *residue form* $C_\Omega(B, D)$). Let **C** be an absolutely irreducible smooth projective algebraic curve of genus g over $\mathbf{F}_q$. Consider an (ordered) set $B = \{P_1, P_2, \cdots, P_n\}$ of distinct $\mathbf{F}_q$ rational points on **C** and a $\mathbf{F}_q$-divisor **D** on **C**.

$$\begin{aligned} &res_B : \Omega(\mathbf{D} - \mathbf{B}) \mapsto \mathbf{F}_q^n \\ &\eta \mapsto (res_{P_1}\eta, \cdots, res_{P_n}\eta) \end{aligned}$$

■

We wish to find the parameters $[n, k, d]$ for this code. Clearly n=dim($\mathbf{F}_q^n$). Note that $\Omega(\mathbf{D}) \subset \Omega(\mathbf{D} - \mathbf{B})$. A differential $\eta \in \Omega(\mathbf{D} - \mathbf{B})$ is send to $[0 \cdots 0] \Leftrightarrow res_{P_i}(\eta) = 0$ for all i, i.e., $\eta \succ \mathbf{D} \Leftrightarrow \eta \in \Omega(\mathbf{D})$. Therefore we have

$$k = i(\mathbf{D} - \mathbf{B}) - i(\mathbf{D})$$

Proposition 5.0.4 *: A geometric Goppa code in residue form defines an $[n, k, d]$ code in the same way as above; n=dim($\mathbf{F}_q^n$),*

$$k = n - (\ell(\mathbf{D}) - \ell(\mathbf{D} - \mathbf{B}))$$

and in particular if $2g - 2 < d(\mathbf{D})$, then

$$k = n - d(\mathbf{D}) + g - 1 + \ell(\mathbf{D} - \mathbf{B})$$

and

$$d \geq d(\mathbf{D}) - 2g + 2$$

Proof. It follows from the previous discussion that

$$k = i(\mathbf{D} - \mathbf{B}) - i(\mathbf{D})$$

It follows from Riemann-Roch theorem that

$$\begin{aligned} i(\mathbf{D}) &= \ell(\mathbf{D}) - d(\mathbf{D}) - 1 + g \\ i(\mathbf{D} - \mathbf{B}) &= \ell(\mathbf{D} - \mathbf{B}) - d(\mathbf{D} - \mathbf{B}) - 1 + g \end{aligned}$$

Therefore we have

$$\begin{aligned} k \quad &= -\ell(\mathbf{D}) + d(\mathbf{D}) + \ell(\mathbf{D} - \mathbf{B}) - d(\mathbf{D} - \mathbf{B})) \\ &= n - (\ell(\mathbf{D}) - \ell(\mathbf{D} - \mathbf{B})) \end{aligned}$$

In case that $2g-2 < d(\mathbf{D})$, we use the corollary after Riemann-Roch theorem to conclude that $i(\mathbf{D}) = 0$ and $\ell(\mathbf{D}) = d(\mathbf{D}) + 1 - g$. Therefore we have

$$\begin{aligned} k &= n - (\ell(\mathbf{D}) - \ell(\mathbf{D} - \mathbf{B})) \\ &= n - d(\mathbf{D}) + g - 1 + \ell(\mathbf{D} - \mathbf{B}) \end{aligned}$$

Let us assume that there are more than $n - (d(\mathbf{D}) - 2g + 2)$ zero-residues for a differential η at $\{P_1, \cdots, P_n\}$ (under the residue map, $res(\eta)$ has more than $n - (d(\mathbf{D}) - 2g + 2)$ zero coordinate, or it has less than $(d(\mathbf{D}) - 2g + 2)$ non-zero corrdinates). We wish to prove that $\eta = 0$.

Let $\mathbf{B}' = \sum_j P_j$ where j runs through all points P_j with $res_{p_j}\eta = 0$. Note that η will have no pole at those P_j. Then we have $d(\mathbf{B}') > n - (d(\mathbf{D}) - 2g + 2)$ or $d(\mathbf{B} - \mathbf{B}') < d(\mathbf{D}) - 2g + 2$. Let us write $\mathbf{D} = \mathbf{D_0} - \mathbf{D}_\infty$ with both effective and disjoint. We have $d(\mathbf{D}) = r - s$. Then we have $(\eta)_0 - (\eta)_\infty \succ \mathbf{D}_0 - \mathbf{D}_\infty - \mathbf{B}$, $d((\eta)_0) \geq d(\mathbf{D}_0) = r, \mathbf{D}_\infty + (\mathbf{B} - \mathbf{B}') \succ (\eta)_\infty$, and $d((\eta)_\infty < s + (d(\mathbf{D}) - 2g + 2) = r - 2g + 2$. Therefore we have $d((\eta)) = d((\eta)_0) - d((\eta)_\infty) > 2g - 2$. If $\eta \neq 0$, then $d((\eta)) = 2g - 2$. Hence $\eta = 0$. What we have proved is that any non-zero differential η can not have more than $n - (d(\mathbf{D}) - 2g + 2)$ zero-residues. Therefore it has more than or equal to $(d(\mathbf{D}) - 2g + 2)$ non-zero-residues. In other words, we prove

$$d \geq d(\mathbf{D}) - 2g + 2$$

■

Remark 3: The lower bounds for the rank and the minimal distance are called the **designed rank** and **designed minimal distance**.

■

Example 3: Let the curve $\mathbf{C}$ be the projective line $P^1_{F_q}$ over a finite field F_q. Let $\beta_1, \cdots, \beta_n$ be n distinct elements in F_q (Note that $n \leq q$). Let $g(x)$ be a polynomial with $g(\beta_i) \neq 0$ for all i. Let us consider P_i equals the point defined by $x - \beta_i = 0$ and P_∞ equals the point at ∞. Let

$$D = \sum_{Q \ finite} ord_Q(g(x))Q - P_\infty = (g(x))_0 - P_\infty$$

and $B = \sum P_i$. Let us pick any non-zero $\eta \in \Omega(D - B)$. Then

$$\eta = f(x)dx = \frac{r(x)}{s(x)}dx$$

Since η has only poles at finite distance in $\mathbf{B}$, then $s(x) \mid \prod(x - \beta_i)$. We may assume that $s(x) = \prod(x - \beta_i)$. Check the order at P_∞. Note that given $deg(r(x)) = u, deg(s(x)) = n$, the number of poles at ∞ for $r(x)/s(x)$ is $u - n$ and dx has two poles at ∞. Since η has at most one pole at ∞, we have $u - n + 2 \leq 1$ or $u \leq n - 1$ or $\deg(r(x)) \leq n - 1$. So we have by the theory of partial fractions that

$$\eta = f(x)dx = \sum \frac{c_i}{x - \beta_i}dx$$

Furthermore, $ord_Q(r(x)) \geq ord_Q(g(x))$ for all Q. Therefore $g(x) \mid r(x)$. We conclude that $\eta \in \Omega(D-B) \Leftrightarrow \sum \frac{c_i}{x-\beta_i} \equiv 0 \ mod(g(x))$. It is easy to see that this is the **classical Goppa Code**.

■

Definition :(**algebraic geometric code** or **a geometric Goppa code** in **primary form** $C_p(B,D)$). Let **C** be an absolutely irreducible smooth projective algebraic curve of genus g over $\mathbf{F}_q$. Consider an (ordered) set $B = \{P_1, P_2, \cdots, P_n\}$ of distinct $\mathbf{F}_q$ rational points on **C** and an $\mathbf{F}_q$-divisor **D** on **C**. For simplicity let us assume that the support of **D** is disjoint from **B**. The *primary Goppa code* is defined as the set of vectors $[a_1 \cdots, a_n] \in \mathbf{F}_q^n$ such that $\sum a_i f(P_i) = 0$ for all $f \in L(\mathbf{D})$.

■

We wish to establish the relations between the three different versions of *geometric Goppa codes* by proving the following proposition,

Proposition 5.0.5 *: The geometric Goppa code in function form $C_L(B,D)$ is the dual code of the geometric Goppa code in primary form $C_p(B,D)$.*

Proof. It is evident from the definition.

■

Proposition 5.0.6 *: The geometric Goppa code in residue form $C_\Omega(B,D)$ is the geometric Goppa code in primary form $C_p(B,D)$.*

Proof. We will prove the code space of $C_\Omega(B,D) \subset$ the code space of $C_p(B,D)$. Then by dimension argument, since they are both of dimensions $n - (\ell(D) - \ell(D-B))$, they must be equal.

Let $f \in L(D)$ and $\eta \in \Omega(D-B)$. Then $(f) \succ -D$ and $(\eta) \succ D-B$. It follows that $(f\eta) = (f) + (\eta) \succ -B$. In other words, all residues outside $\{P_i\}$ are 0's. So we have

$$\sum f(P_i) res_{P_i}(\eta) = 0$$

Hence we conclude that the code space of $C_\Omega(B,D)$ is in the dual space of the code space of $C_L(B,D)$.

■

Proposition 5.0.7 *: Let C be a smooth projective curve with distinct rational places $\mathbf{P_1}, \cdots, {}_m, P$ and $\mathbf{D} = \sum_\mathbf{i} \mathbf{P_i}$, then there exists a differencial $(\omega) \succ D$ such that*

$$res_{P_i}(\omega) = 1, \ fori = 1, \cdots, m$$

Let $\mathbf{W} = \mathbf{B} + (\omega)$. Clearly the supports of $\mathbf{W}.\mathbf{B}$ are disjoint. Then The geometric Goppa code in residue form $C_\Omega(B,D)$ is isomorphic to the geometric Goppa code in function form $C_L(B, W-D)$.

Proof. Clearly it follows from the **Remark** of Proposition 4.7.4 that the differential ω exists. The divisor $(\omega) = -B + W$, such that the support of $\mathbf{B}$ is disjoint from the support of ($\mathbf{W}$. Let us define a map $\pi : C_L(B, W - D) \mapsto C_\Omega(B, D)$ by

$$\pi(f) = f\omega$$

It is easy to see that it is an isomorphism. ■

This theory is sensitive to the curves, divisors involved. For instance, there are curves with very few rational points or even no rational points. Clearly the theory is bad or void in that case. For non-singular plane curves, since the plane P^2_K has $q^2 + q + 1$ points, there are at most $q^2 + q + 1$ many points on the curve, and the Goppa code will be short. Therefore we shall consider space curves. Or the divisor selected is poor for the application purposes (we have to take the decoding process in consideration). We are interested in special curves with many rational points so that the selection is easy and special divisors which may aid us in decoding.

We will illustrate the above method by several examples as follows.

Example 4: Let us fix the ground field to be $\mathbf{F_{2^4}}$. Let us consider curve C defined by $x_0^5 + x_1^5 + x_2^5$ over $\mathbf{F}_{2^4}$. It follows from **Example 4** of section 5.7 that the number of rational points is 65 and the genus $g = 6$. Let us take one rational point P, $\mathbf{B}$= the sum of the remaining 64 points and $\mathbf{D}$= $37P$. Then $d(\mathbf{D} - \mathbf{B}) = -27 < 0$ and $\ell(\mathbf{D} - \mathbf{B}) = 0$. Note that $10 = 2g - 2 < d(\mathbf{D}) = 37$. Therefore for *Goppa code* in *residue form*, the rank $k = 64 - 37 + 6 - 1 = 32$, and the minimal distance $d \geq 27$. As for *Goppa code* in *function form*, the rank $k = 32$, and the minimal distance $d \geq 27$. We have the following data: $n = 64$, $k = 32$, the rate of information is $k/n = 32/64 = 1/2$, and $(d - 1)/2 \geq 13$. comparing with the **Reed-Solomon** code over F_{2^4}, we have $n = 2^4 - 1 = 15$, $1 \leq k < 15$, say $k = 7$, then the rate of information is closed to 1/2, and $d = n - k + 1 = 9$. It may correct 4 errors. It is a much shorter code with weaker power of correcting power. ■

5.1 Comparisons with Reed-Solomon Code.

This section is après Pretzel, page 69. The commonest used code in industry is the Reed-Solomon code. The advantages of it are the speed of decoding and the easiness to be understood. If the speed is not an issue, then the geometric Goppa code has an advantage of being more accurate which will be illustrated in this section.

Let us consider a Reed-Solomon code over the field F_{2^4} defined by all polynomials of degree < 7. Then it is a $[15, 7, 9]$-code where $15 = 2^4 - 1, 7 = 7, 9 = 15 - 7 + 1$. So the number of message symbols is 7 and it can correct $4 = (9 - 1)/2$ errors. Let us consider a geometric Goppa code based on the curve $x_0^5 + x_1^5 + x_2^5$ (cf **Example 4** of section 5.7). By Plücker's formula, its genus g is 6. We have computed to find that there are 65 rational points (cf **Example 4** of section 4.7). Let one of these 65 points be P. Let us consider a one point code with $\mathbf{B}$ = the sum of other 64 points and $\mathbf{D} = 37P$. Then it is an $[n, k, d]$-code where

$n = 64, k = 32, d \geq 27$. Using SV algorithm, it may correct $(27 - 6 - 1)/2 = 10$ errors. Let us process through the said Reed-Solomon-code four times. Then we process 28 message symbols, while we may process 32 symbols through the said Goppa-code.

Let us compare these two processors. The common yardstick is the failure rates. Namely, the probabilities each one will fail to recover the original messages or simply unable to decode due to more than permissible errors. The first phenomena are due to the more than allowed errors and mistakenly decoding to wrong messages. To simplify our notations, we say that the decoder always return a *error message* in either cases. Let us compute the probability for each processors to return an *error message*,i.e., if there are more than allowed errors appear. For Reed-Solomon process, if there are 5 or more errors, then the processor will not decode and reject the received word. Let the channel has a probability p of being incorrect and q of being correct. Then $p + q = 1$ and

$$1 = 1^n = (p+q)^n = \sum_{i=0}^{n} \mathbf{C}_i^n p^i q^{n-i}$$

for a plain processor without self-correcting capability, the probability of returning an *error message* is

$$r_0 = \sum_{i=1}^{1} p^i = 1 - q$$

For Reed-Solomon processor, the probability of returning an *error message* is

$$r_1 = \sum_{i=5}^{15} \mathbf{C}_i^{15} p^i q^{15-i} = 1 - \sum_{i=0}^{4} \mathbf{C}_i^{15} p^i q^{15-i}$$

By the same arguments, using SV algorithm for Goppa-code, the probability of returning an *error message* is

$$r_2 = \sum_{i=11}^{64} \mathbf{C}_i^{64} p^i q^{64-i} = 1 - \sum_{i=0}^{10} \mathbf{C}_i^{64} p^i q^{64-i}$$

Let $p = 0.01$, and use it for 32 times for a plain processor. Then the above number is $r_0 = 0.2750196640$, $r_1 = 0.27627423 \times 10^{-6}$. Therefore to use the processor four times, the probability R_2 of returning an *error message* is

$$R_2 = 1 - (1 - r_1)^4 = 0.11052 \times 10^{-5}$$

We have $r_2 = 0.595098292 \times 10^{-10}$. Therefore, Goppa-code is more accurate than Reed-Solomon-code. We may consider a block of 70,000 message symbols. We have to use the plain processor 70,000 times, the Reed-Solomon processor 10,000 times, the Goppa-code 2,188 times. The probabilities of *error message* for these three processors are

$$\begin{aligned}
1 - (1 - r_0)^{70000} &= 100\% \; for\ plain\ processor \\
1 - (1 - r_1)^{10000} &= 0.27591868\% \; for\ Reed - Solomon \\
1 - (1 - r_2)^{2188} &= 0.00002188\% \; for\ Goppa.
\end{aligned}$$

Later, we will discuss the *DU* algorithm. Using it, we may may correct up to 13 errors for this particular Goppa-code. Then we have

$$r_3 = \sum_{i=14}^{64} \mathbf{C}_i^{64} p^i q^{64-i} = 1 - \sum_{i=0}^{13} \mathbf{C}_i^{64} p^i q^{64-i} = 0.1383190860 \times 10^{-10}$$

and $R_3 = 1 - (1 - r_3)^{2188} = 0.000000000655500732\%$ which is much better than Reed-Solomon-code.

5.2 Improvement of Gilbert-Varshamov's bound

It follows from Proposition 1.7.3 (**Shannon's Theorem**) that we have to consider a sequence of codes with lengths tending to ∞ to achieve an asymptotic good result. On the other hand, Proposition 1.8.4 (Gilbert-Varshamov's bound) establishes a lower bound,

$$\alpha(\delta) \geq 1 - H_q(\delta)$$

In this section, we will show that the above Gilbert-Varshamov's bound can be improved by a family of Goppa codes. We will consider Goppa-codes based on a sequence of curves C_ℓ with $\ell \mapsto \infty$. We require that the numbers n_ℓ of rational non-singular points of C_ℓ tends to ∞ to satisfy the requirement of Proposition 1.7.3 (Shannon's Theorem), then it follows from Proposition 4.8.2 that the genus g_ℓ must tends to ∞. Certainly, this consideration is only for theoretical purposes. For any application, we can not allow $\ell \mapsto \infty$.

Modular curves and Shimura modular curves

For the theoretic part of coding theory, we are interested in a sequence of curves. There are well-known sequences of algebraic curves as the *modular curves* which are discussed in this section.

From the *complex analysis*, we view an *elliptic curve* E as a torus, which can be represented as the quotient space of C/L, where the lattice L can be defined as a *rank 2 Z-submodule* of the complex numbers C generated by two complex numbers ω_1, ω_2 such that ω_1/ω_2 is not a real number. We may choose the order of ω_1, ω_2 such that $im(\omega_1/\omega_2) > 0$. Then ω_1', ω_2' define the same lattice L iff

$$\begin{pmatrix} a, & b \\ c, & d \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \end{pmatrix} = \begin{pmatrix} \omega_1' \\ \omega_2' \end{pmatrix}$$

where the two by two matrix above is in the *full modular group*=the special linear group over $Z = SL_2(Z) = \Gamma$. Let the *principal congruence subgroup* of level N be

$$\Gamma(N) = \{\begin{pmatrix} a, & b \\ c, & d \end{pmatrix} : a \equiv d \equiv 1, b \equiv c \equiv 0 \ mod \ N\}$$

It is easy to see that $\Gamma(1) = \Gamma$.

Two lattices L, L' induce isomorphic elliptic curves iff $L = c * L'$ where $c \in C^* = C\backslash\{0\}$. Then we may normalize the pair (ω_1, ω_2) to $(z, 1) = (\omega_1/\omega_2, 1)$. Due to our convention of ordering, we have $im(z) > 0$. We have the following computation

$$z' = \frac{\omega_1'}{\omega_2'} = \frac{a\omega_1 + b\omega_2}{c\omega_1 + d\omega_2} = \frac{az+b}{cz+d}$$

We shall define

$$\alpha(z) = \begin{pmatrix} a, & b \\ c, & d \end{pmatrix} z = \frac{az+b}{cz+d}$$

Let $\mathcal{H} = \{\text{z} \in \mathcal{C} : \text{im (z)} > 0\}$ be the upper halfplane of the complex plan. It is easy to see that (recall ad-bc=1),

$$im(z') = im(\frac{az+b}{cz+d}) = im(\frac{(az+b)(c\bar{z}+d)}{|cz+d|^2}) = im(\frac{adz+bc\bar{z}}{|cz+d|^2}) = \frac{im(z)}{|cz+d|^2}$$

Therefore α sends elements in $\mathcal{H}$ to elements in $\mathcal{H}$. Let $Y(N) = \Gamma(N)\backslash\mathcal{H}$ and endow it with the quotient topology. It is known that $Y(N)$ is an affine curve. It is possible to deduce that $Y(1)$ is the *moduli curve* of all isomorphic classes of elliptic curves. Let $X(N)$ be the compactification of $Y(N)$. Then it is well-known that $X(N)$ is a projective algebraic curve, and the sequence of $X(N)$ is a sequence of curves.

Very similar to the modular curves we have the Shimura curves or Shimura modular curves, we shall only mention them in the simplist way. What we will use in the coding theory is the following Proposition 5.2.1.

Given a finite field F_{p^2}. Let $\ell(\neq p)$ be a prime. Let $\Gamma_0(\ell)$ denote the following multiplicative subgroup of $SL_2(Z)$

$$\Gamma_0(\ell) = \{\begin{pmatrix} a, & b \\ c, & d \end{pmatrix} : c \equiv 0 \ mod \ \ell\}$$

Then $\Gamma_0(\ell)$ acts on the halfplane $\mathcal{H}$ and $\Gamma_0(\ell)\backslash\mathcal{H}$ is an affine curve $Y_0(\ell)$ (for a detailed discussion, see Tsfasman, Vlăduţ, and Zink[33]). We define the reduction of $Y_0(\ell)$ by the characteristic p and complete it to a projective curve $X_0(\ell)(= C_\ell)$. Then its genus is $\lfloor \ell/12 \rfloor$. We have the following

Proposition 5.2.1 *(Ihara-Tsfasman-Vlăduţ- Zink): For a finite field F_{p^2} with p prime, there exists a sequence of curves C_ℓ(= Shimura modular curves $X_0(\ell)$) which are with genus $g_\ell = \lfloor \ell/12 \rfloor$ and $n_\ell \geq (p-1)(\ell+1)/12$ rational points. Thus $g_\ell/n_\ell \leq 1/(p-1)$.*

Proof. Ihara[24], Tsfasman-Vlăduţ- Zink[33]. ∎

Remark: Since a non-singular plane curve has at most $q^2 + q + 1$ rational points, then the Shimura curves $X_0(\ell)$ do not have a non-singular planar model if $\ell > (12(p^4+p^2+1)/(p-1)-1$.

■

The Theorem of Tsfasman, Vlăduţ, and Zink

Recall the following *entropy* function $H_q(x)$ with $H_q(0) = 0$ and defined for $0 < x < (q-1)/q$

$$H_q(x) = xlog_q(q-1) - xlog_q(x) - (1-x)log_q(1-x)$$

We have the following Proposition,

Proposition 5.2.2 . *The maximum value of $H_q(x) - x$ is attained for $x = (q-1)/(2q-1)$. Its value is $log_q(2q-1) - 1$.*

Proof. Using Calculus, we differentiate $H_q(x) - x$ and set it to 0. We have

$$log_q(q-1) - log_q(x) - log_q\ e + log_q(1-x) + log_q\ e - 1 = 0$$

The above equation can be rewritten as

$$log_q(q-1)(1-x)/(qx) = 0$$

or

$$(q-1)(1-x)/(qx) = 1$$

Solving the above equation, we get

$$x = (q-1)/(2q-1)$$

and

$$\begin{aligned}&(q-1)/(2q-1)log_q(q-1) - (q-1)/2q-1)log_q(q-1)/(2q-1)\\&\quad -(1-(q-1)/(2q-1))log_q(1-(q-1)/(2q-1))\\&= log_q(2q-1) - 1.\end{aligned}$$

Using the second derivative test, we get,

$$-\frac{log_q\ e}{x} - \frac{log_q\ e}{1-x} < 0 \text{ if } 0 < x < (q-1)/q$$

we find it is negative, so we have the maximal point. ■

We have the following theorem for coding theory.

Theorem 5.2.3 *: Let us use the notations of the previous Proposition 5.2.1. Moreover, we assume that $p \geq 7$. Then we have a one-point code on $X_0(\ell)$ with block length $n_\ell - 1$ which tending to ∞ such that for $\delta = (q-1)/(2q-1)$ the relative minimum distance $d_\ell/(n_\ell - 1)$ tends to a limit $> \delta$ and their rate of information $k_\ell/(n_\ell - 1)$ tends to a limit $> 1 - H_q(\delta)$.*

Proof. Let us consider the sequence of Shimura curves $X_0(\ell)$. Let us select one rational point P and call the other rational points $P_1, \cdots, P_{n_\ell - 1}$. Let $\mathbf{B} = \sum_{i=1}^{n_\ell - 1} P_i$. Let $\mathbf{D} = \alpha_\ell P$ with $\alpha_\ell > 2g_\ell - 2$. Then the rank k_ℓ and minimal distance d_ℓ satisfies

begineqnarray* k$_\ell \geq (n_\ell - 1) - \alpha_\ell + g_\ell$
$d_\ell \geq \alpha_\ell - 2g_\ell + 2$and the rate of distance δ_ℓ, the rate of information R_ℓ, are given as

$$R_\ell = k_\ell/(n_\ell - 1) \geq (n_\ell - \alpha_\ell + g_\ell - 1)/(n_\ell - 1)$$
$$\delta_\ell = d_\ell/(n_\ell - 1) \geq (\alpha_\ell - 2g_\ell + 2)/(n_\ell - 1)$$

Choose α_ℓ so that $(\alpha_\ell - 2g_\ell + 2)/(n_\ell - 1) \mapsto \delta$. Then the limit of δ_ℓ is at least δ and the limit of rate of information R_ℓ is at least

$$1 - \delta - lim(g_\ell/n_\ell) \geq 1 - \delta - 1/(p-1)$$

Under our assumption $p \geq 7$, for $p = 7$, we have $log_{7^2}(2 * 49 - 1) = 1.1755 > 7/6 = 7/(7-1)$ or the inequality $log_{p^2}(2p^2 - 1) - p/(p-1) > 0$ is satisfied. It is not hard to see that it is a monotonic increasing function, therefore we always have $log_{p^2}(2p^2 - 1) > p/(p-1)$ for all $p \geq 7$. It follows from the preceding Proposition that

$$Lim_{\ell \mapsto \infty} R_\ell = 1 - \delta - lim(g_\ell/n_\ell) \geq 1 - \delta - 1/(p-1) > 1 - \delta - log_q(2q-1) = 1 - H_q(\delta)$$

■

Exercises

(1): Find the generator matrix for the code defined in **Example 2**.
(2): Find the check matrix for the code defined in **Example 2**.
(3): Write a computer program to code any message for the code defined in **Example 2**.

Chapter 6

Decoding Geometric Goppa Codes

6.1 Introduction

As we pointed out in the previous section, the three different forms of *geometric Goppa codes* determine each other. In the present section, we will discuss only the decoding procedures for the *primary form.* Recall that the primary Goppa code is defined as the set of vectors $[a_1, \cdots, a_n] \in F_q^n$ such that $\sum a_i f(P_i) = 0$ for all $f \in L(D)$. Note that the only check we have to carry out for a vector $a = [a_1, \cdots, a_n]$ to be a code word is to check if $\sum a_i \phi_j(P_i) = 0$ for a basis $\{\phi_j\}$ of the vector space $L(D)$. Later on when we mention **check**, we either mean to use the preceding way of check or use a **check matrix** to check. Then the decoding processes for the other two forms naturally follows.

The theoretical constructions of codes will be of interest to the general public if there is an economical way of decoding them. We will discuss the possible ways.

Most ways of decoding just use matrix theory, hence are easy. There are two prominent ways, among many other methods, of decoding which are the Skorohogatov-Vlăduţ algorithm (SV algorithm) (1991) and the Duursma algorithm (DU algorithm) (1993) which adopts the scheme of Feng and Rao of *majority voting* (1993).

The shortcomings of the above mentioned two methods are the slow processes of solving linear equations. In general, they require n^3 (if we factor in some extra burden) steps to solve the system of linear equations which is comparably slower than the process of decoding Reed-Solomon code (n^2 steps). However, we already know that the *geometric Goppa codes* are more accurate than the *Reed-Solomon* codes (cf last chapter).

The decoding procedures of correcting t errors are consist in general of three steps:

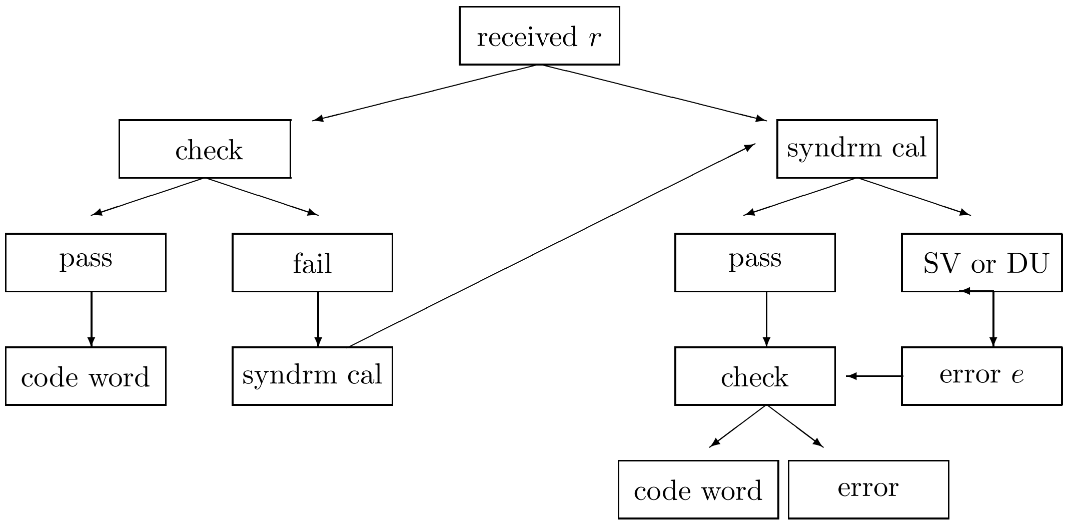
(1): The sender and the receiver agree on the *generator matrix* and the *check matrix* (cf Exercises 4 and 5), the receiver will use the *check matrix* or the **check** procedure to find if the received message is correct. If it is correct, then go to the next block of message. If not, then use the following *syndrome calculation.*

Note that the following *syndrome calculation* can be applied independent of the preceding procedure of using the *check matrix.* Either the received word r will pass the test of **Symdrome** or fail. If it passes, and neither the *check matrix* nor the **check** procedure has been used yet, then further test using the above ‘procedure or the check matrix will decide if it is a code vector (no error) or there are more than t errors (return a message "error"). If it fails the *syndrome calculation*, then we will go to the next step (2).

(2): We start either the SV algorithm (§ 6.3) or DU algorithm (§ 6.4), which is with the basic *assumption* that there are t or less errors. At the end of the procedures, we construct an error vector e.

(3): Further test will decide if $r - e$ is a code vector. If it is, then we complete the decoding procedure and correct the errors (occasionally, it may correct more than t errors) successively. If it is not, then there are more than t errors and return a message "error".

We will have the following flowchart:



Flowchart 2

In the above *Flowchart 2*, *syndrm cal* means *syndrome calculation* (see below), *error e* means *error vector e*, and *error* means *error message*, i.e., there are more than t errors and hence the decoder fails. If the word r already goes through the **check** procedure, and fail. Let it goes through the **syndrom cal**. If it pass, since it already fail the check, then it goes directly to **error**.

In any cases, if it goes to **syndrm cal** and fail, then it should goes to the right hand side of **SV or DU**.

The left hand side of the above flowchart is routine once the sender and the receiver agree on the same check matrixi if it is called for. We will concentrate on the right hand side of the above flowchart, especially the step of using SV algorithm (cf § 6.3) or DU algorithm (cf § 6.4) to produce an error vector e.

6.2 Error-Locator

Let us consider a primary code $C_p(\mathbf{B}, \mathbf{D})$ based on a curve C of genus g where $\mathbf{B} = P_1 + P_2 + \cdots + P_n$ is a sum of distinct rational points and $\mathbf{D}$ is a divisor with the support of $\mathbf{D}$ disjoint from $\mathbf{B}$. Let a word c be transmitted and a word $r = c + e$ is received. The word e is called the *error* word. The decoding process is to find e, given only r. Then we find $c = r - e$.

Recall that one way of the **check** is to compute $\sum \phi_j(P_i)r_i$ for $\{\phi_j\}$ a basis for $L(D)$. This number $\phi(P_i)r_i$ is our main tool of searching. We define,

Definition : For any rational function $\phi \in$ the rational function field of C with no pole at $\mathbf{B}$, we define $\cdot$ product as

$$\phi \cdot r = \sum_j \phi(P_j)r_j$$

where $r_j = r(P_j)$ and call it the *syndrome* of r with respect to ϕ. Note that it is a number. Let $\mathbf{W}$ be a divisor. We say that r has no *syndrome* with respect to $L(\mathbf{W})$, if $\phi \cdot r = 0$ for all $\phi \in L(\mathbf{W})$. In other words, if there is no syndrme with respect to $L(D)$, then r is a code word. We define the $\odot$ product as

$$\phi \odot r = [\phi(P_1)r_1, \cdots, \phi(P_n)r_n]$$

which is a word. ∎

We first define an **error locator** theoretically, and then find the properties of it, and use the properties to locate it. Once we get the error locator, we will use it to find all error locations and the values at them for the error vector e. In this way, we will find the original code word $c = r - e$. It will be a great help if we know all locations of the error vector e. Even a partial knowledge of the error locations will be meaningful. We define

Definition : Let $e = (e_1, \cdots, e_n)$ be an error word. The point P_i is called an **error location** (with respect to e) if $e_i \neq 0$. A non-zero function θ is called an **error locator** (with respect to e) if $\theta(P_i) = 0$ for all error locations P_i and θ has no poles among $P_1, \cdots, P_n$. Thus $\theta \odot e = 0$ always. ∎

From an error locator θ we can not determine all error locations M of e since it may be zero outside M. However, if the set $M' = \{P_i : \theta(P_i) = 0\}$ is not too big comparing with M (we have $M' \supset M$), then we will show later that it is possible to determine M and all coordinates $\{e_i : e_i \neq 0\}$, and solve the decoding problem.

Let us consider $L(\mathbf{D})$. Let $\{\varphi_1, \cdots, \varphi_m\}$ be a basis of $L(\mathbf{D})$. Then we may write $\{\varphi_i(P_j)r_j\}$ as a $m \times n$ matrix $(\varphi_i(P_j)r_j)$. Note that if every row adds up to 0, then $\varphi_i \cdot r = 0, \forall i$, so there is no *syndrome* with respeect to $L(D)$ and $r = c$ or $e = 0$. In general, we will use a simpler way to decide if a received word r has no syndrome as follows,

Proposition 6.2.1 : *Consider a received word r with error word e having weight $\leq t$. If for all functions χ in $L(\mathbf{Y})$ with $d(\mathbf{Y}) \geq t + 2g - 1$ and the support(Y) disjoints from $\mathbf{B}$ $\chi = 0$, then $e = 0$, i.e., there is no syndrome.*

Proof. Note that e is a code word in $C_p(\mathbf{B}, \mathbf{Y})$ which can be identified with the residue form $C_\Omega(B, Y)$ (Proposition 4.0.5). Furthermore it follows from Proposition 5.0.2 that the minimal distance $d \geq d(Y) - 2g + 2 = t + 1$. Then $e = 0$. ∎

Syndrome Calculations

We have the following Proposition for *syndrome calculations*.

Proposition 6.2.2 *: Consider a received word r with error word e having weight $\leq t$. If all functions χ in $L(\mathbf{W})$ with $d(\mathbf{W}) \geq t + 2g - 1$ and $\mathbf{D} \succ \mathbf{W}$ satisfy $\chi \cdot r = 0$, then $e = 0$, i.e., there is no syndrome and $r \in$ the code space.*

Proof. Since $\mathbf{D} \succ \mathbf{W}$, then $L(\mathbf{W}) \subset L(\mathbf{D})$, and $\chi \cdot c = 0$ since $\chi \cdot c = 0$ for all $\chi \in L(D)$. Therefore $\chi \cdot r = \chi \cdot e = 0$ for all $\chi \in L(\mathbf{W})$. We conclude that $e = 0$ by the previous Proposition. ∎

It follows from the preceding Proposition that if we only consider a smaller divisor W with $D \succ W$, then as long the received word r takes the following **syndrome calculas**, the results are either r has more than t errors or r is a code word. The *syndrome calculation* is the computation of the following matrix equation with $m + 1 = \ell$ $(\mathbf{W})$, (sometimes we may assume that $d(\mathbf{W}) = 2g + t - 1$ as in the case of one-point codes).

$$\begin{bmatrix} \varphi_0(P_1), & \cdots & \cdots, & \varphi_0(P_n) \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \varphi_m(P_1), & \cdots & \cdots, & \varphi_m(P_n) \end{bmatrix} \cdot \begin{bmatrix} r_1 \\ \cdot \\ \cdot \\ r_n \end{bmatrix} = \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ 0 \end{bmatrix} ?$$

Indeed we have (1) If the answer is the zero vector 0, then there is no *syndrome* with respect to $L(\mathbf{W})$. However we can only conclude that either r is a code word or there are more than t errors. If we have used the **check** or the check matrix to decide that there are errors, then certainly there are more than t errors. Otherwise we have to further test to distinguish those two cases, either we return to the check matrix or we have to go through the **check** for all $\varphi \in L(\mathbf{D})$, i,e., we check for the vector $[\cdots, \varphi_i \cdot r, \cdots]$ for a basis $\{\varphi_i\}$ of $L(\mathbf{D})$. If the answer is a zero vector, then r is a code word, if the answer is a non-zero vector, then there are more than t errors.

Let us go back to the *syndrome calculation.* (2) If the answer is a non-zero vector, we will *assume* that there are fewer than t errors and use the *syndrome table* to find an error locator and the error vector e (see below). Certainly, we have to check if $r - e$ is a code word by using the check matrix or the **check** procedure. If $r - e$ is a code word, then we succeed in correcting the errors. If $r - e$ is not a code word, then there are more than t errors and the decoder fails. We need $d(\mathbf{W}) \times n$ (sometimes only $(2g + t - 1) \times n$ as in the case of one-point codes) multiplications for the *syndrome calculation* of every block.

Suppose an **error locator** θ is found. We wish to estimate the size of the set $M' = \{P_i : \theta(P_i) = 0\}$.

Proposition 6.2.3 : *Let* $\mathbf{A}$ *be a divisor. Then any function* $\theta \in L(\mathbf{A})$ *can have at most* $d(\mathbf{A})$ *points as zeroes outside the support of* $\mathbf{A}$.

Proof. Let us write the divisor $(\theta)+\mathbf{A}$ as $\sum_{Q_i \in support\ of\ \mathbf{A}} n_i Q_i + \sum_{Q_j \notin support\ of\ \mathbf{A}} n_j Q_j \equiv \mathbf{E}+\mathbf{F}$. If $\mathbf{F} \not\succ 0$ or $\mathbf{E} \not\succ 0$. then $\mathbf{E}+\mathbf{F} \not\succ 0$. It is impossible. Suppose $\mathbf{F} \succ 0$, $\mathbf{E} \succ 0$ and $d(\mathbf{F}) > d(\mathbf{A})$. Then $d(\mathbf{E}+\mathbf{F}) > d(\mathbf{A})$, and $d((\theta)+\mathbf{A}) = d(\mathbf{A}) > d(\mathbf{A})$. A contradiction.

■

6.3 SV Algorithm

Let us consider a *geometric Goppa code in primary form* $\mathbf{C}_p(\mathbf{B})$ based on a curve C with genus g and a suitable divisor $\mathbf{D}$. Let us assume that the minimal distance is d and the number of errors t is with $t \leq \lfloor \frac{d-g-1}{2} \rfloor$. The SV algorithm requires many numerical inequalities. They are for (1) syndrome calculation to decide if the syndrome of $r = 0$. (2) the existence of error locator θ. (3) the usage of θ to compute the error vector e.

Sufficient Relations

We want to find auxiliary divisors $\mathbf{W}$, $\mathbf{A}$, $\mathbf{Y}$, $\mathbf{X}$ which satisfiy the **list of relations** between them: (1). $\mathbf{D} \succ \mathbf{W}$, (2). $\mathbf{D} \succ \mathbf{Y}+\mathbf{A} \succ \mathbf{X}$ and (3). $d(\mathbf{W}) \geq t+2g-1$, (4). $\ell(\mathbf{A}) > t$, (5). $d(\mathbf{A}) < d(\mathbf{D}) - (2g-2) - t$, (6). $d(\mathbf{X}) \geq d(\mathbf{A}) + 2g - 1$, (7).$d(\mathbf{Y}) \geq t + 2g - 1$ and (8). support($\mathbf{Y}$)$\cap\mathbf{B}$ =support($\mathbf{A}$)$\cap\mathbf{B}$=$\emptyset$.

By the preceding Proposition 6.2.2, $\mathbf{W}$, if exists, can be used for *syndrome* calculation. Do we know if those divisors $\mathbf{D}$, $\mathbf{W}$, $\mathbf{A}$, $\mathbf{Y}$, $\mathbf{X}$ exist? Let consider a simple example of one point code, afterall we often use an one point code.

Example 1: Let us consider one point codes of a special kind. Let us consider a one point code $\mathbf{C}_p(\mathbf{B}, m\mathbf{P})$ where $m > 2g-2$. we take $\mathbf{D} = mP$, $\mathbf{A} = bP$ with $0 \leq b \leq m$, $\mathbf{W} = \mathbf{Y} = (m-b)P$ and $\mathbf{X} = mP = \mathbf{D}$. It is easy to check that for the above **list of relations** (1). $\mathbf{D} \succ \mathbf{W}$, (2). $\mathbf{D} \succ \mathbf{Y}+\mathbf{A} \succ \mathbf{X}$ and (8). support($\mathbf{Y}$)$\cap\mathbf{B} = \emptyset$. We have to select m, b suitably so that all numerical conditions are satisfied. First we will deduce several numerical relations.

Lemma: If $0 \leq t$, (t is the number of errors we wish to correct) and $d(\mathbf{D}) \geq 3g+2t-1$, then there is a positive integer b, such that

$$t+g \leq b \leq d(\mathbf{D}) - 2g - t + 1 \tag{1}$$

Furthermore, since the divisor $\mathbf{A}$ is with $d(\mathbf{A}) = b$, then

$$(1) : \ell(\mathbf{A}) > t$$
$$(2) : b = d(\mathbf{A}) < d(\mathbf{D}) - (2g-2) - t = d(\mathbf{D}) - 2g - t + 2$$
$$(3) : d(\mathbf{D}) - b \geq 2g + t - 1 > 2g - 2$$

Note that the *designed minimal distance* $d = d(\mathbf{D}) - 2g + 2$. The above equations mean $t \leq \lfloor \frac{d-g-1}{2} \rfloor$.

Proof: Clearly

$$2t \leq d(\mathbf{D}) - 3g + 1 \Leftrightarrow t + g \leq d(\mathbf{D}) - 2g - t + 1 = d - t - 1$$

Then we just pick up an integer b in between. Furthermore, by Riemann's Theorem, we have

$$\ell(\mathbf{A}) \geq d(\mathbf{A}) + 1 - g > t$$

The parts (2) and (3) are obvious. ■

Now going back to Example 1, the conditions (1),(2) and (8) on the **list of relations** are satisfied. For (3), d($\mathbf{W}$)=m-b=d($\mathbf{D}$)-d($\mathbf{A}$)≥ t+2g-1. For (4), it follows from (1) of the above lemma. For (5), it follows from (2) of the above lemma. For (6), it follows from (2) of the above lemma. For (7), it follows from (2) of the above lemma. It follows that all numerical conditions are satisfied. So in this case there does exist those divisors. ■

Syndrome table

Let $\{\phi_1, \cdots, \phi_u\}$ be a basis for $L(\mathbf{X})$, let $\{\psi_1, \cdots, \psi_v\}$ be a basis for $L(\mathbf{A})$, let $\{\chi_1, \cdots, \chi_w\}$ be a basis for $L(\mathbf{Y})$. Note that $v = \ell(\mathbf{A}) > t$. Without losing generality, we will use the notation that

$$S_{i,j} = S_{ij}(e) = \psi_i \chi_j \cdot e$$

for the error word e. Note that we let the received word be $r = c + e$ with c the original code word and e the error vector, then we have, since $\mathbf{D} \succ \mathbf{A} + \mathbf{Y} \succ \mathbf{X}$ and $S_{ij} = \psi_i \chi_j \in L(\mathbf{A} + \mathbf{Y}) \subset L(\mathbf{D})$. Therefore we have $S_{ij}(c) = 0$ and

$$S_{i,j} = S_{ij}(e) = S_{ij}(r) - S_{ij}(c) = S_{ij}(r)$$

In other word, although e is to be found, we know $S_{ij}(e)$ by computing $S_{ij}(r)$. We will form a $v \times w$ matrix $[S_{i,j}]$ and call it the *syndrome table*.

Theoretical properties of error locators

The following is a useful and important criterion for error locator θ,

Proposition 6.3.1 *: Assume that $d(\mathbf{Y}) \geq t + 2g - 1$ and support($\mathbf{Y}$) is disjoint from $\mathbf{B}$. Let e be an error word with weight at most t. Let θ be a rational function without poles in $\mathbf{B}$. Then θ is an error locator for $e \Leftrightarrow \theta\chi \cdot e = 0$ for all $\chi \in L(\mathbf{Y})$.*

Proof. Let $e' = \theta \odot e = (\theta(P_1)e_1, \cdots, \theta(P_n)e_n)$. Note that weight of $e' \leq t$.

($\Rightarrow$) Since θ is an error locator, then we have $e' = 0$. We have

$$\theta\chi \cdot e = \chi\theta \cdot e = \chi \cdot e' = 0$$

for all $\chi \in L(\mathbf{Y})$.

($\Leftarrow$) Since $\chi \cdot e' = 0$ for all $\chi \in L(\mathbf{Y})$, then e' is a code word in $C_p(\mathbf{B}, \mathbf{Y})$ which has a minimal distance $\geq d(\mathbf{Y}) - 2g + 2 \geq (2g + t - 1) - 2g + 2 = t + 1 > t$. Therefore $e' = 0$ and

θ is an error locator for e.

■

Construction of error locators

The main reference is the Proposition 6.3.1. Now we will construct an error locator using linear algebra with the help of the preceding Proposition.

Proposition 6.3.2 : *The following system of equations has a non-trivial solution* $[\alpha'_1, \cdots, \alpha'_v]$,

$$\sum_i^v S_{ij}(r)\alpha_i = 0, \qquad j \le w \qquad (2)$$

which means in the matrix form,

$$\begin{bmatrix} \alpha_1, \cdots\cdots, \alpha_v \end{bmatrix} \cdot \begin{bmatrix} S_{1,1}, & \cdots & \cdots, & S_{1,w} \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ S_{v,1}, & \cdots & \cdots, & S_{v,w} \end{bmatrix} = \begin{bmatrix} 0, \cdots\cdots, 0 \end{bmatrix}$$

The left null space of the above matrix is not trivial. For any non-trial solution $[\alpha'_1, \cdots, \alpha'_v]$, *let* $\theta = \sum \alpha'_i \psi_i$. *Then* θ *is an* **error locator** *for* e *in* $L(\mathbf{A})$.

Proof. From our assumption, we conclude that $\ell(\mathbf{A}) = v > t$. Let M be the set of error locations of e. Let us consider the following system of equations,

$$\sum_i \psi_i(P_k)\alpha_i = 0, \qquad P_k \in M$$

There are t equations in v variables α_i and $v > t$. Hence there must be a non-trivial solution $[\alpha'_1, \cdots, \alpha'_v]$. Let $\theta = \sum \psi_i \alpha'_i$. Then $\theta(P_k) = 0$ for all $P_k \in M$. Therefore θ is an error locator for e. We will show that the system of equations

$$\sum_i S_{ij}(r)\alpha'_i = 0, \qquad j \le w$$

is satisfied. It follows from the preceding Proposition that $\sum_i S_{ij}(r)\alpha'_i = \sum_i S_{ij}(e)\alpha'_i = (\sum \alpha'_i \psi_i)\chi_j \cdot e = \theta\chi_j \cdot e = 0$, since θ is an error locator of e.

The only thing we want to verify is that the set $\{\alpha'_1, \cdots, \alpha'_v\}$ satisfies equation (2). It follows from Proposition 6.3.1 that $\theta\chi \cdot e = 0$ for all $\chi \in L(\mathbf{Y})$, especially, for all $\chi_j \in L(\mathbf{Y})$. Therefore we have,

$$\sum_i S_{ij}(r)\alpha'_i = \sum_i S_{ij}(e)\alpha'_i = \theta\chi_j \cdot e = 0$$

for all j. It follows from Proposition 6.3.1 that θ is an error locator for e in $L(\mathbf{A})$. For any non-trivial solution $[\alpha'_1, \cdots, \alpha'_v]$, let $\theta = \sum \alpha'_i \psi_i$. Then the above equation is satisfied, and θ is an error locator.

■

Finding the error vector e

The following Proposition tells us how to use an error locator θ to find e.

Proposition 6.3.3 *: Let θ be an error locator (of e) $\in L(\mathbf{A})$ and M' be the set of positions i such that $\theta(P_i) = 0$. Then we have (1): the cardinal number of M', $||M'||, \leq d(\mathbf{A})$. (2): Recall $\{\phi_1, \cdots, \phi_u\}$ is a basis of $L(\mathbf{X}) \subset L(\mathbf{D})$. Moreover, with $d(\mathbf{X}) \geq d(\mathbf{A}) + 2g - 1$, if we solve the following system of equations,*

$$\sum_{P_k \in M'} \phi_i(P_k) E_k = \phi_i \cdot r \qquad i = 1, \cdots, u \qquad (3)$$

where E_k are all indeterminates, then the solution exists uniquely. (3): Furthermore, the error vector $e = [e_1, \cdots, e_n]$ is defined by $\{e_k\}_{P_k \in M'}$ and $e_j = 0$, for all $j \notin M'$.

Proof. The first statement follows from Proposition 6.2.3, since $M' \subset B$ and support$(A) \cap B = \emptyset$. Note that $L(\mathbf{X}) \subset L(\mathbf{D})$, and $\phi_i \cdot r = \phi_i \cdot e$. We may replace all $\phi_i \cdot r$ in the above equation by $\phi_i \cdot e$. Let $M =$ all error locations of e, we have $M \subset M'$. We have

$$\sum_{k \in M'} \phi_i(P_k) E_k = \phi_i \cdot e = \phi_i \cdot r.$$

Since $\phi_i(P_j) = 0$ for all $j \notin M \subset M'$, clearly e is a solution. Let e', e^* be two non-trivial solutions. Then their difference $e' - e^*$ satisfies the following system of equations,

$$\phi_i \cdot (e' - e^*) = 0, \qquad i = 1, \cdots u$$

which means that $(e' - e^*) \in C_p(\mathbf{B}, \mathbf{X})$, which has a minimal distance $d = d(\mathbf{X}) - 2g + 2 \geq d(\mathbf{A}) + 1$. On the other hand, e', e^* have only zero values outside M' which has a cardinality of at most $d(\mathbf{A})$. We conclude that $e' - e^* = 0$ or $e' = e^*$. ∎

The above system of equations (3) can be written in the matrix form with $i_k \in M'$ as

$$\begin{bmatrix} \phi_1(P_{i_1}), & \cdots & \cdots, & \phi_1(P_{i_m}) \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \phi_u(P_{i_1}), & \cdots & \cdots, & \phi_u(P_{i_m}) \end{bmatrix} \cdot \begin{bmatrix} E_{i_1} \\ \cdot \\ \cdot \\ E_{i_m} \end{bmatrix} = \begin{bmatrix} \phi_1 \cdot r \\ \cdot \\ \cdot \\ \phi_u \cdot r \end{bmatrix}$$

Let $E_{i_1} = e_{i_1}, \cdots, E_{i_m} = e_{i_m}$ be a set of solutions of the preceding equation. At this stage, we have to further check if $c = r - e$ is a code word. We may use the *check matrix* or the following procedure. Let $\ell = \ell(\mathbf{D})$ and $\{\phi_1, \cdots, \phi_u, \cdots, \phi_\ell\}$ be a basis of $L(\mathbf{D})$. Let us consider the following matrix equation,

$$\begin{bmatrix} \phi_{u+1}(P_{i_1}), & \cdots & \cdots, & \phi_{u+1}(P_{i_m}) \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \phi_\ell(P_{i_1}), & \cdots & \cdots, & \phi_\ell(P_{i_m}) \end{bmatrix} \cdot \begin{bmatrix} e_{i_1} \\ \cdot \\ \cdot \\ e_{i_m} \end{bmatrix} = \begin{bmatrix} \phi_{u+1} \cdot r \\ \cdot \\ \cdot \\ \phi_\ell \cdot r \end{bmatrix}$$

If the above matrix equation is satisfied, then $\phi_i \cdot c = \phi_i \cdot (r - e) = 0$, for all i, and c is a code word. Otherwise, the decoding procedure fails, and there are more than t errors.

Summary

On a curve C of genus g. Let us consider a *Goppa code in primary form* $\mathbf{C}_p(\mathbf{B},\mathbf{D})$ for some suitable divisor $\mathbf{D}$. Let the minimal distance $d = d(\mathbf{D}) - 2g + 2$ and let $t = \lfloor \frac{d-g-1}{2} \rfloor$ be the number of errors to be corrected. Let us select auxiliary divisors $\mathbf{W}$, $\mathbf{A}$, $\mathbf{Y}$, $\mathbf{X}$. The relation between the divisors are $\mathbf{D} \succ \mathbf{W}$, $\mathbf{D} \succ \mathbf{Y} + \mathbf{A} \succ \mathbf{X}$ and $\ell(\mathbf{W}) \geq 2g + t - 1$, $\ell(\mathbf{A}) > t$, $d(\mathbf{A}) < d(\mathbf{D}) - (2g-2) - t$, $d(\mathbf{X}) \geq d(\mathbf{A}) + 2g - 1, d(\mathbf{Y}) \geq t + 2g - 1$, support$(\mathbf{Y}) \cap \mathbf{B} = \emptyset$.

For one point code of the special kind, we may select $m \geq 3g + 2t - 1$ and b with $t + g \leq b \leq m - 2g - t + 1$ (according to Lemma of § 6.3, such b exists). Let P be a rational point disjoint from $\mathbf{B}$, $\mathbf{D}$=mP, $\mathbf{A}$=bP, $\mathbf{W}$=$\mathbf{Y}$=$(m-b)P$, $\mathbf{X}$=$\mathbf{D}$. It is easy to see that all numerical inequalities of this section are satisfied. This SV algorithm can decode up to $\lfloor \frac{d-g-1}{2} \rfloor$ errors, with a speed depending on the speed of solving a system of linear equations.

If we receive a word r, we use the agreed *check matrix* oor the **check** procedure to test if r is a code word. If it is not a code word, then using $L(\mathbf{W})$, we go through the *syndrome calculation* (Proposition 6.2.2). Or we may go to *syndrome calculation* directly without using the *check matrix* oor the bf check procedure first. If the *syndromes* with respect to $L(\mathbf{W})$ are 0's, then either r is a code word or there are more than t errors. A further test according to Proposition 6.2.2 will distinguish those two situations. If the *syndromes* with respect to $L(\mathbf{W})$ are not all 0's, solve the system of equations (2) to find an error locator θ. Using the error locator θ to determine the possible set of errors M' (cf Proposition 6.2.3) (which may have more than all error locations of e while with a cardinality $\leq d(\mathbf{A})$). Then we solve the system of equations (3) to find the error vector e. At this stage, we have to further check if $c = r - e$ is a code word. We may use the *check matrix* or let $\ell = \ell(\mathbf{D})$ and $\{\phi_1, \cdots, \phi_u, \cdots, \phi_\ell\}$ be a basis of $L(\mathbf{D})$. Let us consider the following matrix equation,

$$\begin{bmatrix} \phi_{u+1}(P_{i_1}), & \cdots & \cdots, & \phi_{u+1}(P_{i_m}) \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \phi_\ell(P_{i_1}), & \cdots & \cdots, & \phi_\ell(P_{i_m}) \end{bmatrix} \cdot \begin{bmatrix} e_{i_1} \\ \cdot \\ \cdot \\ e_{i_m} \end{bmatrix} = \begin{bmatrix} \phi_{u+1} \cdot r \\ \cdot \\ \cdot \\ \phi_\ell \cdot r \end{bmatrix}$$

If the above matrix equation is satisfied, then $\phi \cdot c = \phi_i \cdot (r - e) = 0$, for all i, and c is a code word. Otherwise, the **decoding procedure** fails, and there are more than t errors.

■

Let us consider the following Example.

Example 2: Let us consider the Klein quartic curve over $\mathbf{F}_{2^4}$ defined by the equation $x^3y + y^3 + x = 0$. Its genus is $g = 3$. Let us take $t = 3$, $b = 6$, $m = 14$. Then all numerical conditions are satisfied.

Pre-computation

(1): Let us count the number of rational points. There are 17 rational points $\{P_1, P_2, \cdots, P_16, P\}$ over $\mathbf{F}_{2^4}$. (cf **Example 4** of section 4.8. We shall use the notations of it). We take $\mathbf{B} = P_1 + \cdots + P_{16}$ and $\mathbf{D} = 14P$ where $P =$ the origin.

(2): We take $\mathbf{A} = 6P$, $\mathbf{W} = \mathbf{Y} = 8P$, $\mathbf{X} = 14P$. It is easy to check that $d(\mathbf{D}) = 14, d(\mathbf{A}) = a = 6, d(\mathbf{Y}) = 8, d(\mathbf{X}) = 14$ and $\mathbf{D} \succ \mathbf{W}, \mathbf{D} \succ \mathbf{A} + \mathbf{Y} \succ \mathbf{X}$.

(3): By direct computation, we know the *rank* of this code is $n - d(\mathbf{D}) + g - 1 = 4$ (cf Exercise 4 and 5), and the **designed minimal distance** $\geq d(\mathbf{D}) - 2g + 2 = 10$. The SV algorithm will correct $\lfloor \frac{d-g-1}{2} \rfloor \geq 3$ errors.

(4): We will compute a basis of $L(\mathbf{D}) = L(\mathbf{14P})$. It is easy to see that the following $\{f_0, f_3, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}, f_{13}, f_{14}\}$ form a basis,

$$\begin{array}{cccccccccccc} f_0 & f_3 & f_5 & f_6 & f_7 & f_8 & f_9 & f_{10} & f_{11} & f_{12} & f_{13} & f_{14} \\ 1 & \frac{1}{x} & \frac{y}{x^2} & \frac{1}{x^2} & \frac{y^2}{x^3} & \frac{y}{x^3} & \frac{1}{x^3} & \frac{y^2}{x^4} & \frac{y}{x^4} & \frac{1}{x^4} & \frac{y^2}{x^5} & \frac{y}{x^5}. \end{array}$$

Note that $ord_P(x) = 3, ord_P(y) = 1$ and $ord_P(f_i) = -i$. The reasons that they form a basis are (1) $f_i \in L(\mathbf{14P})$, (2) they are linearly independent over $\mathbf{F}_{2^2}$, (3) by Riemann's theorem, $\ell(\mathbf{14P}) = 14 + 1 - g = 12$.

We shall compute the following 12×16 matrix C,

$$C = \begin{bmatrix} f_0(P_1) & ,\cdots & \cdots & ,f_0(P_{16}) \\ f_3(P_1) & ,\cdots & \cdots & ,f_3(P_{16}) \\ \cdots & \cdots & \cdots & \cdots \\ f_{14}(P_1) & ,\cdots & \cdots & ,f_{14}(P_{16}) \end{bmatrix}$$

It is easy to see that the first 4, f_0, f_3, f_5, f_6, of $\{f_i\}$ form a basis for $L(\mathbf{A})$ and the first 6, $f_0, f_3, f_5, f_6, f_7, f_8$, of $\{f_i\}$ form a basis for $L(\mathbf{Y}) = L(\mathbf{W})$.

We shall further compute $f_i f_j$ for $f_i \in L(\mathbf{A})$ and $f_j \in L(\mathbf{Y})$. We have the following relations,

$$f_i f_j = f_{i+j} \ for \ (i,j) \neq (5,7)$$
$$f_5 f_7 = f_{12} + f_5$$

We will compute the *generator matrix* and the *check matrix* which are 16×4 and 4×16 matrices respectively (cf Exercise 4, 5).

Syndrome calculation

Let us consider the received word r, code word c, error word e. We compute $f_i \cdot r$ for $f_i \in L(\mathbf{Y}) = L(8P)$, i.e.,

$$\begin{bmatrix} f_0(P_1) & ,\cdots & \cdots & ,f_0(P_{16}) \\ f_3(P_1) & ,\cdots & \cdots & ,f_3(P_{16}) \\ \cdots & \cdots & \cdots & \cdots \\ f_8(P_1) & ,\cdots & \cdots & ,f_8(P_{16}) \end{bmatrix} \cdot \begin{bmatrix} r_1 \\ r_2 \\ \cdot \\ r_{16} \end{bmatrix} = \begin{bmatrix} f_0 \cdot r \\ f_3 \cdot r \\ \cdot \\ f_8 \cdot r \end{bmatrix}$$

If $f_i \cdot r = 0$ for $i = 0, 3, 5, 6, 7, 8$, then either there are more than t errors or r is a code word. The total number of multiplications is $6 \times 16 = 96$. We have to compute the following,

$$\begin{bmatrix} f_9(P_1) & ,\cdots & \cdots & ,f_9(P_{16}) \\ f_{10}(P_1) & ,\cdots & \cdots & ,f_{10}(P_{16}) \\ \cdots & \cdots & \cdots & \cdots \\ f_{14}(P_1) & ,\cdots & \cdots & ,f_{14}(P_{16}) \end{bmatrix} \cdot \begin{bmatrix} r_1 \\ r_2 \\ \cdot \\ r_{16} \end{bmatrix} = \begin{bmatrix} f_9 \cdot r \\ f_{10} \cdot r \\ \cdot \\ f_{14} \cdot r \end{bmatrix}$$

The total number of computations for the case that there is no error or there are more than t errors is $12\times16 = 192$. We shall consider a concrete example. To simplify the notations of the numerical involved in F_{2^4}, let us consider a general number $a\alpha\beta + b\beta + c\alpha + d$ where a, b, c, d are 0 or 1. Let us first represent it as $[abcd]$ a four bits number, and then represent the four bits number $[abcd]$ as an integer $a2^3 + bs^2 + c2 + d$ in the *decimal* expression. So all four bits numbers become integers between 0 and 15. Let us order the points $\{P_i\}$ by the indices. Let the code word c, error word e and received word r be as follows

$$\begin{array}{ccrrrrrrrrrrrrrrrr} \mathrm{c} & = & 12 & 12 & 10 & 9 & 2 & 1 & 13 & 3 & 6 & 9 & 2 & 15 & 1 & 0 & 5 & 0 \\ \mathrm{e} & = & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 14 & 0 & 15 \\ \mathrm{r} & = & 12 & 12 & 10 & 9 & 2 & 1 & 13 & 3 & 6 & 9 & 2 & 15 & 0 & 14 & 5 & 15 \end{array}$$

We will use $L(8P)$ for syndrome calculation. The received word r will not pass the *syndrome* calculation, for instance $f_0 \cdot r = 8 \neq 0$.

Calculation for error locator

The *syndrome table* $[S_{i,j}]$ can be formulated at once as follows,

i\\j	0	3	5	6	7	8
0	8	11	5	15	2	1
3	11	15	1	3	9	7
5	5	1	9	7	11	6
6	15	3	7	14	6	13

We have to solve the following system of equations to find an error locator θ

$$\sum_{i=0,3,5,6} S_{ij}(r)\alpha_i = 0 \qquad j = 0, 3, 5, 6, 7, 8$$

Explicitly, we have

$$\begin{aligned} &8\alpha_0 + 11\alpha_3 + 5\alpha_5 + 15\alpha_6 = 0 \\ &11\alpha_0 + 15\alpha_3 + 1\alpha_5 + 3\alpha_6 = 0 \\ &5\alpha_0 + 1\alpha_3 + 9\alpha_5 + 7\alpha_6 = 0 \\ &15\alpha_0 + 3\alpha_3 + 7\alpha_5 + 14\alpha_6 = 0 \\ &2\alpha_0 + 9\alpha_3 + 11\alpha_5 + 6\alpha_6 = 0 \\ &1\alpha_0 + 7\alpha_3 + 6\alpha_5 + 13\alpha_6 = 0 \end{aligned}$$

According to Proposition 6.3.2, we have to further check the following. After solving the above equation, we have to check if it is a permissible error word by computing the equations below. In the transposed matrix form, it can be written as

$$\begin{bmatrix} 0 & 8 & 4 & 3 \\ 8 & 3 & 8 & 7 \\ 4 & 8 & 11 & 11 \\ 3 & 7 & 11 & 7 \\ 12 & 11 & 3 & 12 \\ 8 & 11 & 12 & 4 \end{bmatrix} \cdot \begin{bmatrix} \alpha_0 \\ \alpha_3 \\ \alpha_5 \\ \alpha_6 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

A non-zero solution is $\alpha_0 = 3, \alpha_3 = 8, \alpha_5 = 6, \alpha_6 = 1$. Note that the multiplication and addition are not the *usual* ones. It must be the ones for the field elements. For instance, $2+3 = \alpha+\alpha+1 = 1 \neq 5$ and $2\times 3 = \alpha \times (\alpha+1) = 1$ instead of 6. The fastest way of solving the small size system of linear equations is still Gaussian elimination. The number of multiplications involved is $6^2 \times 4/3 = 48$. The corresponding error locator is $\theta = 3f_0 + 8f_3 + 6f_5 + f_6$. Let us find the zero set of θ. We use the following table with the rows of f_0, f_3, f_5, f_6 pre-computed (say, we want to compute $f_5(P_6)$, we have $f_5(P_6) = \frac{\alpha\beta+\beta+\alpha+1}{\alpha^2} = \beta + 1 = 5$), and the values of θ is computed by the formula $\theta = 3f_0 + 8f_3 + 6f_5 + f_6$.

	P_1,	P_2,	P_3,	P_4,	P_5,	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	P_{16}
f_0	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1
f_3	0,	0,	3,	2,	3,	3,	2,	2,	14,	13,	10,	8,	15,	12,	9,	11
f_5	0,	0,	1,	1,	4,	5,	6,	7,	12,	15,	9,	11,	5,	4,	7,	6
f_6	0,	0,	2,	3,	2,	2,	3,	3,	8,	10,	14,	13,	9,	11,	12,	15
θ	3,	3,	3,	10,	4,	15,	5,	8,	4,	4,	2,	11,	0,	0,	9,	0

Table of values

To find the zero set, we have only to add the row vectors $[f_i(P_1), \cdots, f_i(P_{16})]$ with the produced coefficients and observe the resulting 0 coordinates. In general, it takes $4 \times 16 = 64$ multiplications to find the zero set of θ. So it is fast. For the present case, the zero set M' is $\{P_{13}, P_{14}, P_{16}\}$.

Finding the error vector e

We have to solve the last set of equations,

$$\begin{aligned}
1E_1 + 1E_2 + 1E_3 &= 0\\
12E_1 + 9E_2 + 11E_3 &= 9\\
4E_1 + 7E_2 + 6E_3 &= 4\\
12E_1 + 3E_2 + 11E_3 &= 2\\
8E_1 + 14E_2 + 13E_3 &= 13\\
13E_1 + 8E_2 + 10E_3 &= 9
\end{aligned}$$

The matrix form of the above system of equations is as follows,

$$\begin{bmatrix} 1 & 1 & 1 \\ 12 & 9 & 11 \\ 4 & 7 & 6 \\ 12 & 3 & 11 \\ 8 & 14 & 13 \\ 13 & 8 & 10 \end{bmatrix} \cdot \begin{bmatrix} E_1 \\ E_2 \\ E_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 9 \\ 4 \\ 2 \\ 13 \\ 9 \end{bmatrix}$$

According to our Proposition 6.3.3, there will be some non-trivial solutions. It is easy to see that $E_1 = 15$, $E_2 = 1, E_3 = 14$ satisfies all equations. For this step, usually it suffices to look at the first 3 equations. The number of multiplications needed is 9.

Recovering the original message

Therefore we conclude,

$$\begin{array}{ccccccccccccccccccc} \mathrm{r} & = & 14 & 14 & 11 & 7 & 12 & 5 & 10 & 7 & 7 & 10 & 9 & 13 & 1 & 15 & 1 & 14 \\ \mathrm{e} & = & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 15 & 1 & 0 & 14 \\ \mathrm{c} & = & 14 & 14 & 11 & 7 & 12 & 5 & 10 & 7 & 7 & 10 & 9 & 13 & 14 & 14 & 1 & 0 \end{array}$$

We have to further check if the code vector c is really a code vector. For that purpose, we check the following system of equations in the matrix form,

$$\begin{bmatrix} 14 & 10 & 8 \\ 6 & 4 & 5 \\ 7 & 5 & 4 \\ 15 & 11 & 9 \\ 9 & 15 & 12 \\ 5 & 6 & 7 \end{bmatrix} \cdot \begin{bmatrix} 15 \\ 1 \\ 14 \end{bmatrix} = \begin{bmatrix} 6 \\ 11 \\ 11 \\ 6 \\ 13 \\ 4 \end{bmatrix}$$

For instance, for the first row, we have

$$\begin{aligned} & 14 \cdot 15 + 10 \cdot 1 + 8 \cdot 14 \\ & = (\alpha\beta + \beta + \alpha)(\alpha\beta + \beta + \alpha + 1) + (\alpha\beta + \alpha) + (\alpha\beta)(\alpha\beta + \beta + \alpha) \\ & = \beta^2 + \alpha^2 + (\alpha\beta + 1)(\beta + \alpha) + \alpha\beta + \alpha\beta + \alpha \\ & = \beta + \alpha = 6 \end{aligned}$$

Indeed all equations are satisfied which implies that $f_i \cdot c = 0$ for all $0 \leq i \leq 14$. We conclude that c is a code word, and we decode successively. Note that if the above equations are not satisfied, than our decoder fails. For this step, it takes 18 multiplications.

The total number of multiplications needed is (1): if there is no error, (A): we use the *check matrix*, then 64, (B): otherwise we use the *syndrome calculation*, then 192. (2): if there are less than t errors, then $192 + 48 + 9 + 64 + 18 = 331$. (3): the decoder fails, then $96 + 48 + 9 + 64 + 18 = 235$. We are processing 4 alphabets which is 16 bits. Per bitwise, we have (1): (A): 4 multiplications, (B): 12 multiplications. (2): $20 \cdot 69$ multiplications. (3): $14 \cdot 75$ multiplications. A modern computer of 500 mhz can correct at least 2 million blocks or 32 million bits per second. ■

Remark: The main mistake of some presentations of SV algorithm is in the *syndrome* calculation. In some books (As in Pretzel **codes and Algebraic Curves** page 66), it is taken to be $S_{ij}(r) = \psi_i\chi_j \cdot r$, if all the precedings are zeroes, then it is claimed that there is no syndrome. We shall look at the following **Example**.

Example 3: Let us consider the Klein quartic curve over $\mathbf{F}_{2^4}$ defined by the equation $x^3y + y^3 + x = 0$. We shall use the notations of **Example 5** of section 5.7. Its genus is $g = 3$.

(1): Let us count the number of rational points. There are 17 rational points, $P_1, P_2, \cdots, P_{16}, P$, over $\mathbf{F}_{2^4}$. Let $\mathbf{B} = P_1 + \sum_{i=3}^{16} P_i$, and $\mathbf{D} = 10P$. We will consider the primary *Goppa* code $C_p(\mathbf{B}, \mathbf{D})$.

(2): We take $t = 1$, $m = 10$, $\mathbf{D} = 10P$, $\mathbf{A} = 5P - P_2$, $\mathbf{Y} = 5P + P_2$, $\mathbf{X} = 10P$. It is easy to check that $d(\mathbf{D}) = 10, d(\mathbf{A}) = b = 4, d(\mathbf{Y}) = 6, d(\mathbf{X}) = 10$, the minimal distance d of $C_p(\mathbf{B}, \mathbf{D}) = C_p(\mathbf{B}, \mathbf{X})$, $d \geq 10 - 4 = 6$, $t \leq \lfloor (d - g - 1)/2 \rfloor \geq 1$, so we may take $t = 1$ and $\mathbf{D} \succ \mathbf{A} + \mathbf{Y} \succ \mathbf{X}$, and the numerical conditions are all satisfied. Let $\mathbf{B} = P_1 + \sum_{i=3}^{i=16} P_i$.

Let the error vector e be $e_i = 0$ for all $i \neq 1$ and $e_1 = 1$. Then clearly, weight of $e = 1$. Let us compute $L(\mathbf{A})$ and $L(\mathbf{Y})$. We have the vector space $L(\mathbf{A}) \subset L(5P)$, and $L(\mathbf{A})$ is generated by $\{\frac{1}{x}, \frac{y}{x^2}\}$. The vector space $L(\mathbf{Y}) \supset L(5P)$, $\ell(\mathbf{Y}) \leq \ell(5P) + 1$ and is generated by $\{1, \frac{1}{x}, \frac{y}{x^2}, \frac{x}{y}\}$. Note that at P_1, $\frac{1}{x}(P_1) = 0 = \frac{y}{x^2}(P_1)$. Therefore, $S_{ij}(e) = \psi_i\chi_j \cdot e = 0$, the *syndrome table* is

$$\begin{bmatrix} 0,0,0,0 \\ 0,0,0,0 \end{bmatrix}$$

a 0 matrix and e is not a code word.

The true *syndrome* calculation is by Corollary **??**. We shall use $L(6P)$ which is generated by $\{1, \frac{1}{x}, \frac{y}{x^2}, \frac{1}{x^2}\}$ and for the function 1, $\sum 1 \cdot e = 1 \neq 0$. Therefore, e is not a code word. We may go through the process to find an error locator. Since the *syndrome table* is a 0 matrix, any non-trivial tuple (α_1', α_2') will do. We may take it to be $(1, 0)$, so $\frac{1}{x}$ is an error locator θ. Eventually, we find e. ■

Exercises

(1): Show that $[1, 1, 1, 2, , 15, 12, , 8, 2, , 8, 6, 14, 11, 0, 1, 0, 0]$ is a code word in **Example 2**.
(2): Show that $[5, 5, 15, 9, 3, 2, 14, 4, 12, 10, 14, 4, 0, 0, 1, 0]$ is a code word in **Example 2**
(3): Show that $[1, 1, 3, 1, 3, 13, 9, 11, 8, 15, 5, 13, 0, 0, 0, 1]$ is a code word in **Example 2**
(4): Find a *generator* matrix for the code in **Example 2**.
(5): Find a *check* matrix for the code in **Example 2**.
(6): Write a computer program to decode the code of **Example 2**.

(7): Write down the details of decoding the geometric Goppa code $C_p(\mathbf{B}, 37P)$ based on $x_0^5 + x_1^5 + x_2^5$ with the ground field $\mathbf{F}_{2^4}$ (cf § 6.4).

(8): Write a computer program to decode the code of **Exercise 4**.

6.4 DU Algorithm

Due to the fact that the preceding SV algorithm only decodes up to $\lfloor \frac{d-g-1}{2} \rfloor$ errors instead of the designed number $\lfloor \frac{d-1}{2} \rfloor$ for a *Goppa* code based on a curve C with genus g, and the best *Goppa* codes are based on curves with large genus g, the shortcoming is serious sometimes. Feng and Rao found a way to get around it by *majority voting*. Duursma follows their idea to extend SV algorithm to DU algorithm which will correct errors up to the designed power of $\lfloor \frac{d-1}{2} \rfloor$.

DU algorithm is more complicate than SV algorithm.

6.5 The details of the DU algorithm

Pre-computation

Let us consider $C_p(\mathbf{B}, \mathbf{D})$, where $\mathbf{B} = P_1 + P_2 + \cdots P_n$, we assume that there is a further rational point P which is distinct from all P_i. Let $\mathbf{D}$ be a divisor with $d(\mathbf{D}) \geq (2g + 2t - 1)$ (we may take $\mathbf{D} = d(\mathbf{D})P$). Let $\mathbf{D}' = \mathbf{D} - (d(\mathbf{D}) - 2g - 2t + 1)P$ (if $\mathbf{D} = d(\mathbf{D})P$, then $\mathbf{D}' = (2g+2t-1)P$). Then we have $d(\mathbf{D}') = 2g+2t-1$ and $\mathbf{D} \succ \mathbf{D}'$. Therefore $L(\mathbf{D}-\mathbf{B}) \supset L(\mathbf{D}'-\mathbf{B})$, and every code word in $C_p(\mathbf{B}, \mathbf{D})$ is a code word in $C_p(\mathbf{B}, \mathbf{D}')$. **We will decode words in $C_p(\mathbf{B}, \mathbf{D}')$.** One way to think about it is the sender and the receiver should agree on the divisor **D'** instead of $\mathbf{D}$ at the beginning. The minimal distance is $\lfloor \frac{2t+1}{2} \rfloor$ and the *DU* algorithm will correct t errors (see below).

Now we shall choose an auxiliary divisor $\mathbf{A}$ with $d(\mathbf{A}) = t$ and with support disjoint from $\mathbf{B}$ (say, $\mathbf{A} = tP$).

The secondary auxiliary divisor we will use is $\mathbf{A}'$ where $\mathbf{A}'$ is defined by $\mathbf{A}' = \mathbf{D}' - \mathbf{A} - (2g-1)P$ (if $\mathbf{A} = tP$ and $\mathbf{D} = d(\mathbf{D})P$, then $\mathbf{A}' = tP = \mathbf{A}$). Then $d(\mathbf{A}') = t$. Note that the important divisor $\mathbf{D}' = \mathbf{A} + \mathbf{A}' + (2g-1)P$. We shall study $L(\mathbf{A} + iP)$, $L(\mathbf{A}' + iP)$ and $L(\mathbf{A} + \mathbf{A}' + iP)$ (see below).

The *syndrome* calculation will not be affected, we just compute $\phi_i \cdot r$ where ϕ_i runs through a basis of $L(\mathbf{D}') = L(\mathbf{A} + \mathbf{A}' + (2g-1)P)$ and r is the received word. The problem is to find an error locator θ.

Ordering of basis

For the convenience of ordering the basis of $L(\mathbf{A} + (2g-1)P)$ or $L(\mathbf{A}' + (2g-1)P)$, we shall generalize the classic orderings for Weierstrass gaps. Let $\mathbf{A_0} = \mathbf{A} - tP$. Then $d(\mathbf{A_0}) = 0$ and $\mathbf{A} + (2g-1)P = \mathbf{A_0} + (2g+t-1)P$. We have

Proposition 6.5.1 *: We have $\ell(\mathbf{A_0} + (2g + t - 1)P) = g + t$. We may select a basis $\{\psi_i\}$ for $L(\mathbf{A_0} + (2g + t - 1)P)$, where $0 \leq i \leq 2g + t - 1$ with g of them missing (so there are precisely $g+t$ of them) such that $\psi_i \in L(\mathbf{A_0}+iP)\backslash L(\mathbf{A_0}+(i-1)P)$, for $0 \leq i \leq (2g+t-1)$.*

Proof. The first part of the Proposition follows from Proposition 4.6.3, since for any canonical divisor $\mathbf{W}$, $\mathbf{W}-(\mathbf{A_0}+(2g+t-1)P)$ is a negative divisor. The other part of the Proposition follows from Proposition 4.6.1, since $\psi_i \in L(\mathbf{A_0}+iP)$ and $\ell(\mathbf{A_0}+iP)-\ell(\mathbf{A_0}+(i-1)P) \leq 1$.

■

Definition : For any rational function $\theta \in L(\mathbf{A_0}+(2g+t-1)P)$, we define $\nu_{\mathbf{A}}(\theta) = min\{i : \theta \in L(\mathbf{A_0} + iP)\}$ and call the g missing i the *Weierstrass gaps* of $\mathbf{A}$. ■

Proposition 6.5.2 *: For a rational function $\theta \in L(\mathbf{A_0} + (2g + t - 1)P)$ with $\theta = b_i\psi_i + \sum_{k<i} b_k\psi_k$ and $b_i \neq 0$, we have $\mu_{\mathbf{A}}(\theta) = i$.*

Proof. . It follows from the previous Proposition. ■

Remark: The classic ordering and the one point code mentioned above in the parenthesis are the ones with $\mathbf{A_0} = \mathbf{0}$ and $\nu_{\mathbf{A}}(\psi_i) = -ord_P(\psi_i)$. ■

Recall that if $\mathbf{A} = \sum A(Q_i)Q_i$, then $A(Q_i)$ is the corresponding coefficient of Q_i. We have the following Proposition.

Proposition 6.5.3 *: For any rational function $\theta \in L(\mathbf{A_0}+(2g+t-1)P) = L(\mathbf{A}+(2g-1)P)$, we have $\nu_{\mathbf{A}}(\theta) = d(\mathbf{A})-\mathbf{A}(P)-ord_P(\theta)$ and given $\nu_{\mathbf{A}}(\theta) = i$, $\nu_{\mathbf{A'}}(\theta') = j$, and $\nu_{\mathbf{A}+\mathbf{A'}}(\theta\theta') = i + j$.*

Proof. Let $\mathbf{A} = \mathbf{E} + \mathbf{A}(P)P = \mathbf{E} + sP$. Then $\mathbf{A_0} = \mathbf{E} + (s - t)P$ and we have

$$\begin{aligned}
&\theta \in L(\mathbf{A_0} + iP) \\
&\Leftrightarrow \theta \in L(\mathbf{E} + (i + s - t)P) \\
&\Leftrightarrow -ord_P(\theta) \leq (i + s - t) \\
&\Leftrightarrow -ord_P(\theta) - s + t \leq i
\end{aligned}$$

Apparently the minimal possible for the last inequality is $-ord_P(\theta) - s + t = d(\mathbf{A}) - \mathbf{A}(P) - ord_P(\theta)$.

Furthermore, if $\theta \in L(\mathbf{A}_0+iP)\backslash L(\mathbf{A}_0+(i-1)P)$, and $\theta' \in L(\mathbf{A'}_0+jP)\backslash L(\mathbf{A'}_0+(j-1)P)$, then $\theta\theta' \in L(\mathbf{A}_0 + \mathbf{A'}_0 + (i + j)P)\backslash L(\mathbf{A}_0 + \mathbf{A'}_0 + (i + j - 1)P)$. ■

Similarly we shall construct bases $\{\chi_i\}$, $\{\phi_i\}$ for $L(\mathbf{A'}+(2g-1)P)$ and $L(\mathbf{A}+\mathbf{A'}+(3g-1)P)$ respectively. Note that $\mathbf{D'} = \mathbf{A}+\mathbf{A'}+(2g-1)P$, therefore we construct more ϕ_i than needed for coding purposes. We will pre-compute the following relations,

$$\psi_i\chi_j = a_{i,j}\phi_{i+j} + \sum_{k<i+j} b_{i,j,k}\phi_k$$

where $a_{i,j} \neq 0$, without losing generality, we may assume $a_{i,j} = 1$.

Syndrome Table

Let $\{\phi_i : 0 \leq i \leq (2g + 2t - 1)\} = L(\mathbf{D}')$. We will construct the following *syndrome table* which is a $(g + t) \times (g + t)$ matrix S. Customarily, we use the (i, j) entry (where i, j are not Weierstrass gaps) to denote

$$S_{i,j} = \psi_i \chi_j \cdot e$$

Note that they are not the usual subindices, in every row and column there are g indices missing. Since we do not know e, the above matrix only exists theoretically. However, if $0 \leq i, j \leq 2g + t - 1$ and $i + j \leq 2g + 2t - 1$, then

$$S_{i,j} = \psi_i \chi_j \cdot e = \psi_i \chi_j \cdot r$$

which is computable and *known.* For $i + j > 2g + 2t - 1$, the terms $S_{i,j}$ are unknown. We will put all $S_{i,j}$, known or unknown, in a matrix form, and it is called the *syndrome table.* It is of the following $(g + t) \times (g + t)$ square form with $*$ known numbers, ? unknown numbers,

$$\overbrace{\begin{bmatrix} * & , & \cdots & \cdots, & *, & * \\ * & , & \cdots & \cdots, & *, & * \\ * & , & \cdots & \cdots, & *, & * \\ * & , & \cdots & \cdots, & *, & * \\ * & , & \cdots & \cdots, & *, & ? \\ * & , & \cdots & * \ , & ?, & ? \\ * & , & \cdots * & ? \ , & ?, & ? \end{bmatrix}}^{g+t}$$

The usages of the syndrome table. Step 0

To begin with, we know the *syndrome table* $\{S_{i,j}\}$ for all $0 \leq i, j \leq 2g + t - 1$, $i + j \leq 2g + 2t - 1$. So we know a part of this *syndrome table* bounded by a line $i + j = 2g + 2t - 1$. With this meager information of knowing only the first t rows and t columns completely, we may be able to find an error locator θ, i.e., solving the following system of equations for $\{\alpha_i\}$, say the non-zero solutions form a set $\{\alpha_i'\}$.

$$\sum_{i \leq t} S_{i,j} \alpha_i = 0, \qquad j \leq 2g + t - 1 \tag{2'}$$

and let $\theta = \sum \alpha_i' \psi_i$. Note that we *assume* that there is a non-zero solution $\{\alpha_i'\}$ which may not be true. Similarly, we may solve the system of equations columnwise to try to find an error locator if we can. If we can find an error locator θ this way (or rowwise), then we may use the following Proposition to solve the decoding problem.

Proposition 6.5.4 *: Let the set $M' = \{P_i : \theta(P_i) = 0\}$. Then its cardinality is at most t, and the following system of equations,*

$$\sum_{P_k \in M'} \phi_i(P_k) E_k = \phi_i \cdot r (= \phi_i \cdot e), \qquad i \leq 2g + 2t - 1 \tag{3''}$$

has an unique non-zero solution $e\mid_{M'}$, where $e\mid_{M'}$ is the restriction of e to the coordinates determined by M'. Note that outside the positions determined by M', e has coordinates zeroes, so we can find e by the preceding information.

Proof. Since $\theta \in L(\mathbf{A})$, the it follows from Proposition 6.2.3 that the cardinality of $M' \le d(\mathbf{A}) \le t$.

Clearly $e\mid_{M'} = e'$ is a solution of $(3'')$. Let e^* be another solution. Then the weight of $e' - e^* \le t$ and $e' - e^*$ will be a solution of the homogeneous system of equations

$$\sum_{P_k \in M'} \phi_i(P_k)E_k = 0, \qquad i \le 2g+2t-1 \tag{5'}$$

In other words, $e' - e^* \in C_p(\mathbf{B}, \mathbf{A}+\mathbf{A}'+(2g-1)P)$. While $C_p(\mathbf{B}, \mathbf{A}+\mathbf{A}'+(2g-1)P))$ has a minimal distance $\ge 2g+2t-1-(2g-2) = 2t+1$. Therefore $e'-e^* = 0$ and the solution is unique. ■

The usages of syndrome table. Step $s+1 < g$

In general, we shall build the *syndrome table* line by line, for all i, j with $0 \le i, j \le 2g+t-1$ from the line $i+j = 2g+2t-1+s$ to the next line $i+j = 2g+2t-1+s+1$ for $s = 0, 1, \cdots, g-1$. Every time after we construct $S_{i,j}$ (see the next section) for all i, j with $0 \le i, j \le 2g+t-1$, $i+j = 2g+2t-1+s+1$, we try to find an error locator θ rowwise or columnwise. There are two possibilities: either we find one (then we precede to decode) or we can not find one (then we push to find the $S_{i,j}$ on the next line). Suppose that we find one, say we find a non-zero solution $\{\alpha'_i\}$ of the following system of equations,

$$\sum_{i \le t+s+1} S_{i,j}\alpha_i = 0, \qquad j \le 2g+t-1 \tag{2''}$$

we let $\theta = \sum \alpha'_i \psi_i$, then θ is an error locator. The following Proposition will guarantee that we will be able to decode the message.

Proposition 6.5.5 *: Let the set $M' = \{P_i : \theta(P_i) = 0\}$. Then its cardinality is at most $s+t+1$, and the following system of equations,*

$$\sum_{P_k \in M'} \phi_\ell(P_k)E_k = \phi_\ell \cdot e, \qquad \ell \le 2g+2t+s \tag{3'}$$

has an unique non-zero solution $e\mid_{M'}$, where $e\mid_{M'}$ is the restriction of e to the coordinates determined by M'. Note that outside the positions determined by M', e has coordinates zeroes, so we can find e by the preceding information.

Proof. Since $S_{i,j} = \psi_i\chi_j \cdot e$ are all known for $0 \le i, j \le 2g+t-1$ and $0 \le i+j \le 2g+2t+s$. Furthermore, by our pre-computation, we have

$$\psi_i\chi_j \cdot e = \phi_{i+j} \cdot e + \sum_{k<i+j} b_{i,j,k}\phi_k \cdot e$$

then $\phi_\ell \cdot e$ are all known for $\ell \leq 2g + 2t + s$. Since $\theta \in L(\mathbf{A} + (s+1)P)$, then it follows from Proposition 6.2.3 that the cardinality of $M' \leq d(\mathbf{A} + (s+1)P) = (s + t + 1)$.

Clearly $e \mid_{M'} = e'$ is a solution of $(3')$. Let e^* be another solution. Then the weight of $e' - e^* \leq (g + t)$ and will be a solution of the homogeneous system of equations

$$\sum_{P_k \in M'} \phi_i(P_k)E_k = 0, \qquad i \leq 2g + 2t \qquad (5')$$

In other words, $e' - e^* \in C_p(\mathbf{B}, \mathbf{A} + \mathbf{A}' + (2g + s)P)$. While $C_p(\mathbf{B}, \mathbf{A} + \mathbf{A}' + (2g + s)P)$ has a minimal distance $\geq 2g + 2t + s - (2g - 2) = 2t + s + 2 > t + s + 1$. Therefore $e' - e^* = 0$ and the solution is unique.

■

The usage of syndrome table. Step $s + 1 = g$.

This is the final step in our discussion of the usage of syndrome table. We expect to find the error word e. Due to the restrictions on the indices $0 \leq i, j \leq 2g + t - 1$ and $i + j = 2g + 2t - 1 + (g - 1) + 1 = 3g + 2t - 1$. Therefore, the *syndrome table* can be constructed partially up to the first indices $\leq g + t$ (or up to the second indices $\leq g + t$), and the value $\phi_i \cdot e$ can be found for all $0 \leq i \leq 3g + 2t - 1$, then the following system of equations

$$\sum_{i \leq g+t} S_{i,j}\alpha_i = 0, \qquad j \leq 2g + t - 1 \qquad (2')$$

will have a non-zero solution $\{\alpha'_i\}$ (cf Proposition 6.3.3). Moreover

$$\theta = \sum_{i \leq g+t} \alpha'_i \psi_i$$

will be an error locator. We have the following Proposition,

Proposition 6.5.6 *: Let the set $M' = \{P_i : \theta(P_i) = 0\}$. Then its cardinality is at most $g + t$, and the following system of equations,*

$$\sum_{P_k \in M'} \phi_i(P_k)E_k = \phi_i \cdot e, \qquad i \leq 3g + 2t - 1 \qquad (3')$$

has an unique non-zero solution $e \mid_{M'}$, where $e \mid_{M'}$ is the restriction of e to the coordinates determined by M'. Note that outside the positions determined by M', e has coordinates zeroes, so we can find e by the preceding information.

Proof. Since $\theta \in L(\mathbf{A} + gP)$, the it follows from Proposition 6.2.3 that the cardinality of $M' \leq d(\mathbf{A} + gP) = (g + t)$.

Clearly $e \mid_{M'} = e'$ is a solution of $(3')$. Let e^* be another solution. Then $e' - e^*$ the weight of $e' - e^* \leq (g + t)$ and will be a solution of the homogeneous system of equations

$$\sum_{P_k \in M'} \phi_i(P_k)E_k = 0, \qquad i \leq 3g + 2t - 1 \qquad (5')$$

In other words, $e' - e^* \in C_p(\mathbf{B}, \mathbf{A} + \mathbf{A}' + (3g-1)P)$. While $C_p(\mathbf{B}, \mathbf{A} + \mathbf{A}' + (3g-1)P)$ has a minimal distance $\geq 3g + 2t - 1 - (2g-2) = g + 2t + 1 > g + t$. Therefore $e' - e^* = 0$ and the solution is unique. ■

So the question is how to construct the *syndrome table* $[S_{i,j}]$ line by line without knowing what is e.

6.6 Feng-Rao's Majority Voting

We will make inductions on $s = 0, 1, \cdots, g-1$, we assume that for all i, j with $i + j \leq 2g + 2t - 1 + s$, S_{ij} are known, and furthermore we *assume* that we can not find an error locator θ with $\nu_{\mathbf{A}}(\theta) \leq t + s$ rowwise (nor columnwise), i.e., the following two system of equations have only trivial solutions,

$$\sum_{i \leq t+s} S_{ij}\alpha_i = 0, \qquad j \leq 2g + t - 1 \tag{2*}$$

$$\sum_{j \leq t+s} S_{ij}\beta_j = 0, \qquad i \leq 2g + t - 1 \tag{2**}$$

Let us use the notation $\mathbf{S}|_{i,j}$ to denote the submatrix $[S_{u,v}(e)]$ for $u \leq i, v \leq j$. Now we will study the terms $S_{i,j}$ for $i + j = 2g + 2t - 1 + s + 1$, which means that in the following submatrix, all $*$ terms are *known*, and we want to study the particular unknown term $S_{i,j}$.

$$S_{|i,j} = \begin{bmatrix} * & , & \cdots & \cdots, & * \\ \cdots & & \cdots & \cdots & \cdots \\ * & , & \cdots, & * \;, & * \\ * & , & \cdots, & * \;, & S_{i,j} \end{bmatrix}$$

Relations between all $S_{i,j}$ for $i + j = 2g + 2t - 1 + s + 1$ and fixed s

Let us consider $\{\phi_i : \nu_{\mathbf{A}+\mathbf{A}'}(\phi_i) = i, 0 \leq i \leq 3g + 2t - 1\} \subset L(\mathbf{A} + \mathbf{A}' + (3g-1)P)$. We may express $\psi_i\chi_j$ for $i + j = 2g + 2t - 1 + s + 1$ (note that $s + 1 \leq g$) as follows,

$$\psi_i\chi_j = a_{i,j}\phi_{i+j} + \sum_{k<i+j} b_{i,j,,k}\phi_k \tag{6}$$

where $a_{i,j} \neq 0$, otherwise $\nu_{\mathbf{A}+\mathbf{A}'}(\psi_i\chi_j) < i + j$. Therefore if all $b_{i,j,k}$ and $\phi_k \cdot e$ are known, then the value $\psi_i\chi_j \cdot e$ is uniquely determined by the value of $\phi_{i+j} \cdot e$ and vice versa. Thus all values $S_{i,j}$ with $i + j = 2g + 2t - 1 + s + 1$ determine each other if the value of $\phi_{i+j} \cdot e$ determined is *correct*. During our construction, the values of $\{S_{i,j} : i + j = 2g + 2t - 1 + s + 1\}$ constructed rowwise and columnwise may not be consistent, i.e., they produce inconsistent values for $\phi_{i+j} \cdot e$. The consistent value $\phi_{i+j} \cdot e$ for fixed i, j, correct or not, will be called the *vote* of each individual $S_{i,j}$.

Now we shall search for all possible values $S_{i,j}$ for fixed i, j with with $i + j = 2g + 2t - 1 + s + 1$. Among them we shall determine which ones are the true values. All true values must be unique, exist, and consistent with other ones for different i, j through the common $\phi_{i+j} \cdot e$.

Possible values of $S_{i,j}$ for fixed s, i, j with $i + j = 2g + 2t - 1 + s + 1$

Let us fix i, j with $i + j = 2g + 2t - 1 + s + 1$. In our situation, the matrix $\mathbf{S}|_{i,j}$ is not completely known, i.e., all entries are known except $S_{i,j}$ term. If the last row of $\mathbf{S}|_{i,j-1}$ is a linear combination of the preceding rows, i.e., the following equations can be solved,

$$\sum_{k<i} \alpha_k S_{k,\ell}(e) = S_{i,\ell}(e) \qquad for \qquad \ell < j \tag{7}$$

Let $\{\alpha'_0, \cdots, \alpha'_{i-1}\}$ be a solution, then $S_{ij}(e)$ can be defined as

$$S_{i,j} = \sum_{k<i} \alpha'_k S_{k,j}(e)$$

Similarly, we do it for columnwise. If the term $S_{i,j}$ can not be determined either rowwise or columnwise, or the solutions in either cases are not unique, or if they both exist and unique but they do not agree, then we consider that there is no value and we consider that the *vote* is invalid. If they both exist and unique and agree, then the selection of $S_{i,j}$ will cast a *valid vote*, namely the hypothetical $\phi_{i+j} \cdot e$.

We will have a better understanding of the above procedure by using linear algebra. Let us give $S_{i,j}$ some value to complete the matrix $\mathbf{S}|_{i,j}$.

Proposition 6.6.1 *: Let $\mathbf{S}|_{i,j} = [S_{k,\ell}(e)]_{(k\le i,\ell\le j)}$ be a $(i\times j)$ matrix. Then (1): $rank(\mathbf{S}|_{i-1,j-1}) = rank(\mathbf{S}|_{i,j})$, $\Leftrightarrow$ $S_{i,j}$ casts a valid vote. (2): $rank(\mathbf{S}|_{i-1,j-1}) = rank(\mathbf{S}|_{i-1,j}) = rank(\mathbf{S}|_{i,j-1})$ $\Leftrightarrow$ there is a unique way to fill the value for $S_{i,j}$ so that $rank(\mathbf{S}|_{i-1,j-1}) = rank(\mathbf{S}|_{i,j})$.*

Proof. Firstly, we shall prove the claim (1).

($\Rightarrow$). We have

$$rank(\mathbf{S}|_{i-1,j-1}) \le rank(\mathbf{S}|_{i,j-1}) \le rank(\mathbf{S}|_{i,j})$$

So they must be all equal. Then by Gaussian Elimination to the last row of $\mathbf{S}|_{i,j}$, when the last row of $\mathbf{S}|_{i,j-1}$ becomes 0, the value of $S_{i,j}$ is determined. Hence our assigned arbitrary value for $S_{i,j}$ must be the determined one. Suppose that there are two ways to give two different values β_1, β_2 for $S_{i,j}$. If we assign β_2 to the matrix, then clearly the last row of $\mathbf{S}|_{i,j}$ will becomes $[0, \cdots, 0, a]$ with $a \neq 0$. Then we have

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

It is against our hypothesis. So the value of $S_{i,j}$ is uniquely determined this way. Similarly, its value is uniquely determined columnwise. Let the value of $S_{i,j}$ be b rowwise, and c columnwise. Let us take the value b for $S_{i,j}$ and then apply the columnwise Gaussian Elimination to the last column. We get a new matrix with last row $[0, \cdots, 0, b - c]$. It follows that $b = c$.

($\Leftarrow$) If $S_{i,j}$ casts a *valid vote*, then by Gaussian Elimination to the last row, the last row will become $[0, \cdot, 0]$. Let us then apply Gaussian Elimination to the last column. The last column will be $[0, \cdots, 0]^t$. It is easy to see that

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

Secondly, we shall prove the claim (2).

Using Gaussian elmination to the last row, and add 0 at the last entry of the last row. Then the last row consist of 0's only. Since Gaussian elmination is reversible, thus we determine the value of $S_{i,j}$ uniquely. Now apply Gaussian elmination to the last column, and we have the last column consist of 0's only. So clearly

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

■

We want to illustrate the relation of the rank of $\mathbf{S}|_{i,j}$ and the existence of certain kind error locators. Let us *assume* that the value of $S_{i,j}$ casts a *valid vote*. Let us consider an error locator θ with $\nu_{\mathbf{A}}(\theta) = i$ and $\theta \in L(\mathbf{A_0} + (2g+t-1)P)$. Let

$$\theta = a_i\psi_i + \sum_{k<i} b_k\psi_k$$

where $a_i \neq 0$. As usual we assume it to be 1. It follows from Proposition 6.3.1 that we always have

$$\theta\chi_\ell \cdot e = \chi_\ell \cdot (\theta \odot e) = \chi_\ell \cdot 0 = 0, \ for\ all\ \ell \leq 2g+t-1$$

Now we slightly weaken the conditions in the above equation to the following

$$\theta\chi_\ell \cdot e = \chi_\ell \cdot (\theta \odot e) = \chi_\ell \cdot 0 = 0, \ for\ all\ \ell \leq j$$

or

$$\psi_i\chi_\ell \cdot e + \sum_{k<i} b_k\psi_k\chi_\ell \cdot e = 0, \ for\ all\ \ell \leq j$$

or just change the notations as

$$S_{i,\ell}(e) = \sum_{k<i}(-b_k)S_{k,\ell}(e) = 0, \ for\ all\ \ell \leq j$$

Thus the last row of $\mathbf{S}|_{i,j}$ is a linear combination of the preceding rows. We rewrite the above in the following way: If the last row of $\mathbf{S}|_{i,j}$ is a linear combination of the preceding rows, i.e., solve the following equations,

$$\sum_{k<i} \alpha_k S_{k,\ell}(e) = S_{i,\ell}(e) \qquad for \qquad \ell < j \tag{7}$$

If there is a solution $\{\alpha'_0, \cdots, \alpha'_{i-1}\}$, then $S_{ij}(e)$ must be

$$S_{i,j} = \sum_{k<i} \alpha'_k S_{k,j}(e)$$

Similarly, we do it for columnwise. Now it is clear that the rank of $\mathbf{S}|_{i,j}$ is related to our *voting* procedure.

We shall re-explain the above using linear algebra. Let us consider a non-zero vector $v = [\overbrace{\cdots, b_k, \cdots, 1}^{i}, 0, \cdots, 0]$. Then v is in the left null-space of $S_{|i-1,j} \Leftarrow$ the rational function θ defined by

$$\theta = \psi_i + \sum_{k<i} b_k \psi_k$$

is an error locator with $\nu_{\mathbf{A}}(\theta) = i$.

We have

Proposition 6.6.2 *: If there are error locators θ and θ' with $\nu_{\mathbf{A}}(\theta) = i$, $\theta \in L(\mathbf{A_0} + (2g + t-1)P)$ and $\nu_{\mathbf{A}'}(\theta') = j$, $\theta' \in L(\mathbf{A}'_{\mathbf{0}} + (2g+t-1)P)$, then $S_{i,j}$ casts a valid vote and the vote is correct. Moreover if there is an error locators θ (respectively if there is an error locators θ') with $\nu_{\mathbf{A}}(\theta) = i$, $\theta \in L(\mathbf{A_0}+(2g+t-1)P)$ (respectively $\nu_{\mathbf{A}'}(\theta') = j$, $\theta' \in L(\mathbf{A}'_{\mathbf{0}}+(2g+t-1)P)$), and $S_{i,j}$ casts a valid vote, then the vote is correct.*

Proof. In the above discussion, we show that the equation (7) can be solved and produces a value for $S_{i,j}$. By Gaussian Elimination to the last row, the last row is reduced to $[0, \cdots, 0]$. Now apply the same reasoning to the columns. The last column is $[0, \cdots, 0]^t$. Therefore we have $rank\mathbf{S}|_{i-1,j-1} = rank\mathbf{S}|_{i,j}$. It follows from the preceding Proposition that $S_{i,j}$ casts a *valid vote*. Let

$$\theta = a_i \phi_i + \sum_{k<i} b_k \phi_k$$

where $a_i \neq 0$. As usual we assume it to be 1. Then we have

$$0 = \theta \cdot e = \phi_i \cdot e + \sum_{k<i} b_k \phi_k \cdot e$$

Since all $\phi_k \cdot e$ are correct as we already know, then $\phi_i \cdot e$ must be correct.

Moreover, if there is an error locators θ with $\nu_{\mathbf{A}}(\theta) = i$, $\theta \in L(\mathbf{A_0} + (2g + t - 1)P)$ and $S_{i,j}$ casts a *valid vote*, then repeat the above argument, we conclude that the *vote* must be correct.

■

Majority Voting

As we point out that we do not know the values of $\phi_\ell \cdot e$ for $j < \ell$. Hence we can not test to find $\phi_\ell \cdot e$ directly. We will go down all $S_{i,j}$ for $i + j = 2g + 2t - 1 + s + 1$, and collect all *valid votes*. The following Proposition says that the majority of valid votes is the correct one,

Proposition 6.6.3 *: Let us assume that there is no error locators θ and θ' with $\nu_{\mathbf{A}}(\theta) \leq t+s$, $\theta \in L(\mathbf{A} + (2g-1)P)$ and $\nu_{\mathbf{A}'}(\theta') \leq t+s$, $\theta' \in L(\mathbf{A}' + (2g-1)P)$ then (the number of correct vote)$-$(the number of incorrect vote) $\geq s+1$. In particular, there is at least one correct vote.*

Proof. Note that $i + j = 2g + 2t - 1 + s + 1$ and $i, j \leq 2g + t - 1$. We conclude that $t + s + 1 \leq i, j \leq 2g + t - 1$.

(1). We claim that there are at least g error locators θ with $\nu_{\mathbf{A}}(\theta)$ distinct numbers between $t+s+1$ and $2g+t-1$.

Since there is no error locator θ with $\nu_{\mathbf{A}}(\theta) \leq t+s$, we may count all error locators in $L(\mathbf{A}+(2g-1)P)$. Let the divisor $\mathbf{E}$ be $\sum_{P_i \in M} P_i$, where M is the error locations of e. Clearly an error locator θ is a non-zero element in $L(\mathbf{A}+(2g-1)P-E)$. Note that $d(\mathbf{A}+(2g-1)P-E) = 2g-1$. Therefore it follows from Riemann's theorem that $\ell(\mathbf{A}+(2g-1)P-E) = g$. Let $\{\theta_i\}$ be a basis. We may use linear changing of basis to make them have distinct values $\nu_{\mathbf{A}}(\theta)$.

(2). Let us count the number p of correct *votes* and the number q of incorrect *votes*. According to the preceding Proposition 6.6.3, $p \geq$ the number of pairs (i,j) with both error locators θ, θ' with $\nu_{\mathbf{A}}(\theta) = i$, $\nu_{\mathbf{A}'}(\theta') = j$, and $\theta \in L(\mathbf{A}+(2g-1)P), \theta' \in L(\mathbf{A}'+(2g-1)P)$, and $q \leq$ the number of pairs (i,j) with no error locators with $\nu_{\mathbf{A}}(\theta) = i$, $\theta \in L(\mathbf{A}+(2g-1)P)$ or $\nu_{\mathbf{A}'}(\theta') = j$, $\theta' \in L(\mathbf{A}'+(2g-1)P)$. Let **the set** J be the set of all integers $\{i : t+s+1 \leq i \leq 2g+t-1\}$, I be the subset of J which is the collection of $\{\nu_{\mathbf{A}}(\theta)\}$ of error locators θ. So it follows from (1) that the cardinality of I, $\mid I \mid$, $= g$.

(3). Let $I' = \{(2g+2t-1+s+1) - j : j = \nu_{\mathbf{A}'}(\theta')\} \cap J$. Then by arguments identical to (1) and (2), we have $\mid I' \mid = g$. We have

$$q \leq \mid J \backslash (I \cup I') \mid = 2g - s - 1 - \mid I \cup I' \mid$$

and

$$p \geq \mid I \cap I' \mid$$

So we have

$$p - q \geq \mid I \cap I' \mid + \mid I \cup I' \mid + s + 1 - 2g = \mid I \mid + \mid I' \mid + s + 1 - 2g = s + 1$$

■

After we tally all valid *votes* and find the winner which is the correct one, we will make the *vote unanimous* by changing all incorrect *vote* and invalid votes and hence all incorrect $S_{i,j}$ and non-exist $S_{i,j}$. Once we complete the extra line $S_{i,j}$ with $i+j = 2g+2t_1+s+1$, we shall try to find an error locator with order $t+s+1$. If we can, then we precede to find the error vector e. If not (and *only if not*), we shall try to find the next line.

6.7 Example

Let us consider the **Klein** quartic curve over $\mathbf{F}_{2^4}$ defined by the equation $x^3y+y^3+x = 0$ (cf **Example 4** of section 5.8. We shall use the notations of it). Its genus is $g = 3$. Let us consider a one point code $C_p(\mathbf{B}, \mathbf{D})$ with $\mathbf{B} = P_1 + \cdots + P_{16}$, and $\mathbf{D} = (2g+2t-1)P = 11P$ with $t = 3$.

We take $\mathbf{A} = \mathbf{A}' = 3P$, and $\mathbf{D}' = \mathbf{D} = 11P$. For the *geometric Goppa code* $C_p(\mathbf{B}, \mathbf{D})$, we have the rank $k = n - d(\mathbf{D}) + g - 1 = 7$ and the *designed minimal distance* $d = d(\mathbf{D}) - 2g + 2 = 7$. Use DU algorithm, we expect to correct 3 errors.

We shall make the following pre-computation.

(1): We pre-compute the *generator matrix* and the *check matrix* (cf Exercises 4 and 5).

(2): We will compute a basis of $L((3g+2t-1)P) = L(14P)$. It is easy to see that the following $\{\phi_0, \phi_3, \phi_5, \phi_6, \phi_7, \phi_8, \phi_9, \phi_{10}, \phi_{11}, \phi_{12}, \phi_{13}, \phi_{14}\}$ form a basis,

$$\begin{array}{cccccccccccc} \phi_0 & \phi_3 & \phi_5 & \phi_6 & \phi_7 & \phi_8 & \phi_9 & \phi_{10} & \phi_{11} & \phi_{12} & \phi_{13} & \phi_{14} \\ 1 & \frac{1}{x} & \frac{y}{x^2} & \frac{1}{x^2} & \frac{y^2}{x^3} & \frac{y}{x^3} & \frac{1}{x^3} & \frac{y^2}{x^4} & \frac{y}{x^4} & \frac{1}{x^4} & \frac{y^2}{x^5} & \frac{y}{x^5}. \end{array}$$

where $ord_P(\phi_i) = -i$. The reasons that they form a basis are (1) $\phi_i \in L(\mathbf{14P})$, (2) they are linearly independent over $\mathbf{F}_{2^2}$, (3) by **Riemann's** theorem, $\ell(\mathbf{14P}) = 14 + 1 - g = 12$.

We shall compute the following 12×16 matrix C,

$$C = \begin{bmatrix} \phi_0(P_1) & , \cdots & \cdots & , \phi_0(P_{16}) \\ \phi_3(P_1) & , \cdots & \cdots & , \phi_3(P_{16}) \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{14}(P_1) & , \cdots & \cdots & , \phi_{14}(P_{16}) \end{bmatrix}$$

It is easy to see that the first 6 of $\{\phi_i\}$ form a basis for $L(\mathbf{A})$.

We shall consider another auxiliary divisor $\mathbf{A} + (2g-1)P = (2g+t-1)P = 8P = \mathbf{A}'$. We have to express $\psi_i \chi_j = \sum_{k \le i+j} a_k \phi_k$ as follows,

$$\begin{aligned} &\phi_0 \phi_i = \phi_i, \ for\ i \le 8 \\ &\phi_3 \phi_i = \phi_{i+3}, \ for\ i \le 8 \\ &\phi_5 \phi_i = \phi_{i+5}, \ for\ i \le 8, \ and\ i \ne 7 \\ &\phi_6 \phi_i = \phi_{i+6}, \ for\ i \le 8, \end{aligned}$$

Using the defining equation $y^3 = x^3 y + x$, we deduce the remaining equations as follows,

$$\begin{aligned} \phi_5 \phi_7 &= \frac{y^3}{x^5} = \frac{1}{x^4} + \frac{y}{x^2} = \phi_{12} + \phi_5, \\ \phi_7 \phi_7 &= \frac{y^4}{x^6} = \frac{x^3 y^2 + xy}{x^6} = \frac{y}{x^5} + \frac{y^2}{x^3} = \phi_{14} + \phi_7, \end{aligned}$$

The above are our pre-computations.

Syndrome calculation

Let us consider the received word r. We use the *check matrix* or the **check** procedure to determine if r is a code word. If it is, then we pass to the next block of received message. The computation requires $7 \times 16 = 112$ multiplications. If it is not, then we start the *syndrome calculation* as follows. We compute $\phi_i \cdot r$ for $\phi_i \in L(\mathbf{A}) = L(8P)$, i.e.,

$$\begin{bmatrix} \phi_0(P_1) & , \cdots & \cdots & , \phi_0(P_{16}) \\ \phi_3(P_1) & , \cdots & \cdots & , \phi_3(P_{16}) \\ \cdots & \cdots & \cdots & \cdots \\ \phi_8(P_1) & , \cdots & \cdots & , \phi_8(P_{16}) \end{bmatrix} \cdot \begin{bmatrix} r_1 \\ r_2 \\ \cdot \\ r_{16} \end{bmatrix} = \begin{bmatrix} \phi_1 \cdot r \\ \phi_3 \cdot r \\ \cdot \\ \phi_8 \cdot r \end{bmatrix}$$

If $\phi_i \cdot r = 0$ for all ϕ_i, then r is a received word with more than 3 errors (since it fails the check matrix test, so it can not be a code word), we shall return an *error* message and we pass to the next block of the received message. The total number of multiplications is $6 \times 16 = 96$. If the matrix is not 0 matrix, we have to compute the following,

$$\begin{bmatrix} \phi_9(P_1) & ,\cdots & \cdots & ,\phi_9(P_{16}) \\ \phi_{10}(P_1) & ,\cdots & \cdots & ,\phi_{10}(P_{16}) \\ \phi_{11}(P_1) & ,\cdots & \cdots & ,\phi_{11}(P_{16}) \end{bmatrix} \cdot \begin{bmatrix} r_1 \\ r_2 \\ \cdot \\ r_{16} \end{bmatrix} = \begin{bmatrix} \phi_9 \cdot r \\ \phi_{10} \cdot r \\ \phi_{11} \cdot r \end{bmatrix}$$

The total number of computations for the case that there is an error is $9 \times 16 = 144$.

Syndrome Tables

(2): We will use the same representation of F_{2^4} as in the **Example 4** of section 5.8. Let us consider the situation that the code word c, the error word e the received word r are as follows,

c	=	13	0	14	3	0	10	11	8	1	10	14	12	1	1	1	1
e	=	0	0	0	0	0	0	0	0	0	0	0	0	1	14	0	15
r	=	13	0	14	3	0	10	11	8	1	10	14	12	0	15	1	14

The received word will not pass the *syndrome* calculation. For instance $\phi_3 \cdot r = 2 \neq 0$. So we look at the following *syndrome table*

	ϕ_0,	ϕ_3,	ϕ_5,	ϕ_6,	ϕ_7,	ϕ_8
ϕ_0	0,	8,	4,	3,	12,	8
ϕ_3	8,	3,	8,	7,	11,	11
ϕ_5	4,	8,	11,	11,	$S_{5,7}$,	
ϕ_6	3,	7,	11,	$S_{6,6}$,		
ϕ_7	12,	11,	$S_{7,5}$,			
ϕ_8	8,	11,				

Syndrome table 1

The above *syndrome table* is constructed by the relation $r = c + e$ and $\phi_i\phi_j \cdot r = \phi_i\phi_j \cdot e$ for all $i + j \leq 2g + 2t - 1 = 11$. If there is an error locator θ with $\nu_{\mathbf{A}}(\theta) \leq 3$ and $\theta \in L(8P)$, then the first two rows must be linear dependent. Since they are not, so there is no such θ. In general, to check if two rows of vectors of length 6 are linearly independent, it takes 6 multiplications at most.

We shall construct for $s = 0$ the terms $S_{i,j}$ for $i + j = 2g + 2t - 1 + 0 + 1 = 12$. There are three terms: $\phi_5\phi_7 \cdot e = S_{5,7}$, $\phi_6\phi_6 \cdot e = S_{6,6}$, $\phi_7\phi_5 \cdot e = S_{7,5}$.

Recall that we use the notations of **Example 2** of section 7.3 where the field F_{2^4} is generated by α, β with $\alpha^2 + \alpha + 1 = 0$ and $\beta^2 + \beta + \alpha = 0$ and any integer $a_0 + a_1 2 + a_2 2^2 + a_2^3$

with $0 \leq a_i \leq 1$ represents $a_0 + a_1\alpha + a_2\beta + a_3\alpha\beta$. Let us compute the following (3×3) submatrix $\mathbf{S}|_{5,5}$ of the *Syndrome table 1*,

$$\begin{bmatrix} 0, & 8, & 4 \\ 8, & 3, & 8 \\ 4, & 8, & 11 \end{bmatrix} = \begin{bmatrix} 0, & \alpha\beta, & \beta \\ \alpha\beta, & \alpha+1, & \alpha\beta \\ \beta, & \alpha\beta, & \alpha\beta+\alpha+1 \end{bmatrix}$$

It is easy to see by direct computation that the above matrix is non-singular. Therefore $S_{5,7}, S_{7,5}$ give only non-valid votes. It follows that $S_{6,6} = 7$ is a correct vote and $\phi_{12} \cdot e = 7$ and $S_{5,7} = S_{7,5} = \phi_{12} \cdot e + \phi_5 \cdot e = 7 + 4 = \beta + \alpha + 1 + \beta = 3$. In general, to find if $S_{5,7}$ casts a *valid vote*, we have to check if $rank(S_{|3,7}) = rank(S_{|3,6}) = rank(S_{|5,6})$?. It needs 13 multiplications. If the answer is yes then we can find its value with 2 more multiplications and its *vote*. If the answer is no, we discard it. Since the matrix is always symmetric, we do not have to do anything about $S_{7,5}$. For $S_{6,6}$, a similar argument will give us 16 multiplications to determine if the value constitutes a *valid* vote, and if so the *vote*. Therefore, in general we need 31 multiplications to fill those 3 spots.

The *syndrome table* is as follows,

	ϕ_0,	ϕ_3,	ϕ_5,	ϕ_6,	ϕ_7,	ϕ_8
ϕ_0	0,	8,	4,	3,	12,	8
ϕ_3	8,	3,	8,	7,	11,	11
ϕ_5	4,	8,	11,	11,	3,	$S_{5,8}$
ϕ_6	3,	7,	11,	7,	$S_{6,7}$,	
ϕ_7	12,	11,	3,	$S_{7,6}$,		
ϕ_8	8,	11,	$S_{8,5}$,			

Syndrome table 2

Now since there is no ϕ_4, then there is no error locator θ with $\nu_{\mathbf{A}}(\theta) = 4$. In other words, $\nu_{\mathbf{A}}(\theta) \leq 4 \Leftrightarrow \nu_{\mathbf{A}}(\theta) \leq 3$. From the point of view of linear algebra, there is no $S_{4,8} \cdot e$, or there is no new completed row, so we do not have to work on the linear dependency of completed row vectors. Similarly, the same things hold for column vectors. Then we may try to fill the *syndrome table* for the next step. We shall try to find the correct values of $S_{5,8}, S_{6,7}, S_{7,6}, S_{8,5}$.

Since the matrix $\mathbf{S}|_{5,5}$ is non-singular, $S_{5,8}, S_{8,5}$ cast invalid votes. We have $s + 1 = 2$, so the number of correct votes must be ≥ 2 and $S_{6,7}, S_{7,6}$ both cast correct vote. Then it is routine to find out $S_{6,7} = 12, S_{7,6} = 12$. By correcting all wrong votes and invalid votes, we have $S_{5,8} = 12, S_{8,5} = 12$.

In general, the computations about ranks of various submatrices use $20 + 18 = 30$ multiplications, and to find the values of $S_{5,8}(= S_{8,5}), S_{6,7}(= S_{7,6})$ use $2 + 3 = 5$ multiplications. So totally it needs 35 multiplications.

The *syndrome table* becomes,

	$\phi_0,$	$\phi_3,$	$\phi_5,$	$\phi_6,$	$\phi_7,$	ϕ_8
ϕ_0	0,	8,	4,	3,	12,	8
ϕ_3	8,	3,	8,	7,	11,	11
ϕ_5	4,	8,	11,	11,	3,	12
ϕ_6	3,	7,	11,	7,	12,	$S_{6,8}$
ϕ_7	12,	11,	3,	12,	$S_{7,7},$	
ϕ_8	8,	11,	12,	$S_{8,6},$		

Syndrome table 3

We will check if there is an error locator θ with $\nu_{\mathbf{A}}(\theta) \leq 5$ and $\theta \in L(8P)$, or an error locator θ' with $\nu_{\mathbf{A}'}(\theta') \leq 5$ and $\theta' \in L(8P)$. Now since the matrix $\mathbf{S}_{|5,5}$ is non-singular, we can not find any one.

In general, we have to check if the first 3 rows are linearly dependent. It takes $12+5=17$ multiplications.

So we consider $s=2, s+1=3$. There are three terms to be considered, $S_{6,8}, S_{7,7}, S_{8,6}$. According to Proposition 6.6.3, (correct votes)$-$(incorrect votes) $\geq s+1=3$. So all of them must be correct. On the other hand, $s=2=g-1$ is the end of our induction. So we should make the last computation.

It can be checked that for the matrix $\mathbf{S}|_{6,7}$, with the rows denoted by $R_0.R_3, R_5, R_6$, we have $R_6 = 6R_5 + 8R_3 + 3R_0$. Therefore, $S_{6,8} = 4 = S_{8,6}$ and $S_{7,7} = \phi_{14} \cdot e + \phi_7 \cdot e = 4 + 12 = \alpha + \alpha\beta + \alpha = 8$.

In general, we have to compute the linear relation between the rows $R_0.R_3, R_5, R_6$ of the matrix $\mathbf{S}|_{6,7}$. Using Gaussian elmination, it takes 26 multiplications, and it takes 3 more multiplications to find the value of $S_{6,8}$. So totally it takes 29 multiplications to find all $S_{6,8}, S_{7,7}, S_{8,6}$.

We have the following complete *syndrome table 4*,

	ϕ_0,	ϕ_3,	ϕ_5,	ϕ_6,	ϕ_7,	ϕ_8
ϕ_0	0,	8,	4,	3,	12,	8
ϕ_3	8,	3,	8,	7,	11,	11
ϕ_5	4,	8,	11,	11,	3,	12
ϕ_6	3,	7,	11,	7,	12,	4
ϕ_7	12,	11,	3,	12,	8,	
ϕ_8	8,	11,	12,	4,		

Syndrome table 4

Now $\theta = \phi_6 + 6\phi_5 + 8\phi_3 + 3$ is an error locator with $\nu_{\mathbf{A}}(\theta) \leq 6$ and $\theta \in L(8P)$. Let us find the zero set of θ. We use the following table with the rows of $\phi_0, \phi_3, \phi_5, \phi_6$ pre-computed,

	P_1,	P_2,	P_3,	P_4,	P_5,	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	P_{16}
ϕ_0	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1
ϕ_3	0,	0,	3,	2,	3,	3,	2,	2,	14,	13,	10,	8,	15,	12,	9,	11
ϕ_5	0,	0,	1,	1,	4,	5,	6,	7,	12,	15,	9,	11,	5,	4,	7,	6
ϕ_6	0,	0,	2,	3,	2,	2,	3,	3,	8,	10,	14,	13,	9,	11,	12,	15
θ	3,	3,	3,	10,	11,	13,	9,	15,	8,	3,	13,	9,	0,	0,	9,	0

Table of values

In general, since $\theta(P_i) = \phi_6(P_i) + 6\phi_5(P_i) + 8\phi_3(P_i) + 3$. We will use the pre-computation of $\phi_j(P_i)$ to locate the zeroes of θ. It takes $4 \times 16 = 64$ multiplications. For the present case, the zero set M' is $\{P_{13}, P_{14}, P_{16}\}$.

We have to solve the last set of equations,

$$\begin{aligned}
&1E_1 + 1E_2 + 1E_3 = 0\\
&15E_1 + 12E_2 + 11E_3 = 8\\
&5E_1 + 4E_2 + 6E_3 = 4
\end{aligned}$$

$$9E_1 + 11E_2 + 15E_3 = 3$$
$$10E_1 + 8E_2 + 13E_3 = 12$$
$$14E_1 + 13E_2 + 10E_3 = 8$$

According to our Proposition 6.3.3, there will be some non-trivial solutions. It is easy to see that $E_1 = 1$, $E_2 = 14, E_3 = 15$ satisfies all equations.

In general, there are at most $g + t = 6$ variables and to solve this system of equations we need 72 multiplications.

Therefore we conclude,

$$\begin{array}{cccccccccccccccccc} \text{r} & = & 13 & 0 & 14 & 3 & 0 & 10 & 11 & 8 & 1 & 10 & 14 & 12 & 0 & 15 & 1 & 14 \\ \text{e} & = & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 14 & 0 & 15 \\ \text{c} & = & 13 & 0 & 14 & 3 & 0 & 10 & 11 & 8 & 1 & 10 & 14 & 12 & 1 & 1 & 1 & 1 \end{array}$$

We have to further test to see if c is a code word. We check the following matrix equation

$$\begin{bmatrix} 13, & 14, & 8 \\ 7, & 6, & 5 \\ 6, & 7, & 4 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 14 \\ 15 \end{bmatrix} = \begin{bmatrix} 7 \\ 11 \\ 11 \end{bmatrix}$$

For instance, the first equation is,

$$\begin{aligned} & 13 \cdot 1 + 14^2 + 8 \cdot 15 \\ & = (\alpha\beta + 1) \cdot 1 + (\alpha\beta + \beta + \alpha)^2 + \alpha\beta(\alpha\beta + \beta + \alpha + 1) \\ & = \beta + \alpha + 1 = 7 \end{aligned}$$

Similarly we may verify the other two equations. Since the above equation is satisfied, we know that c is a code word and we successively decode. In general, we need 112 multiplications for an error-free block, 407 multiplications to correct 1 block with fewer than 3 errors and 407 multiplications to return an *error* message (to indicate that there are More than 3 errors). Since it processes 7 alphabets which is 28 bits, per bitwise, it takes 4, 14.5, 14.5 multiplications respectively. It is faster than SV algorithm. ■

Exercises

(1): Show that $[0, 4, 12, 12, 1, 0, 0, 13, 12, 12, 3, 11, 0, 0, 0, 0]$ is a code word in the **Example**.
(2): Show that $[13, 9, 9, 4, 0, 14, 1, 0, 13, 4, 9, 6, 0, 0, 0, 0]$ is a code word in the **Example**.
(3): Show that $[14, 2, 12, 2, 0, 10, 0, 1, 2, 9, 12, 14, 0, 0, 0, 0]$ is a code word in the **Example**.
(4): Show that $[10, 1, 14, 6, 0, 9, 0, 0, 13, 10, 15, 3, 1, 0, 0, 0]$ is a code word in the **Example**.
(5): Show that $[0, 6, 3, 9, 0, 5, 0, 0, 6, 0, 5, 3, 0, 1, 0, 0]$ is a code word in the **Example**.
(6): Show that $[1, 5, 14, 12, 0, 2, 0, 0, 13, 8, 5, 5, 0, 0, 1, 0]$ is a code word in the **Example**.
(7): Show that $[1, 12, 9, 11, 0, 6, 0, 0, 14, 10, 5, 9, 0, 0, 0, 1]$ is a code word in the **Example**.

(8): Find a *generator* matrix for the code in the **Example**.

(9): Find a *check* matrix for the code in the **Example**.

(10): Write a computer program to decode the code of the **Example**.

(11): Using DU algorithm, write down the details of decoding the geometric Goppa code $C_p(\mathbf{B}, 37P)$ based on $x_0^5 + x_1^5 + x_2^5$ with the ground field $\mathbf{F}_{2^4}$ (cf § 6.8).

(12): Write a computer program to decode the code of **Exercise 4**.

APPENDIX

Appendix A

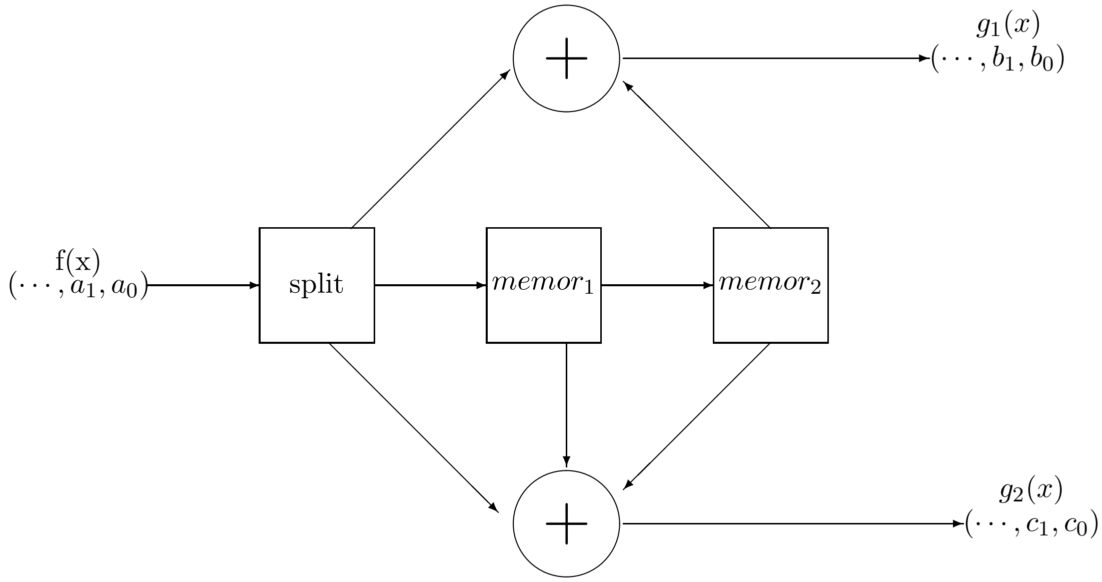
Convolutional Codes

A.1 Representation

The concept of convolution code was introduced by P. Elias ([17]). Any sequence of data or a stream of data $(a_0, a_1, \cdots, a_n, \cdots)$ with $a_i \in \mathbf{F}$ can be represented as a power series,

$$f(x) = \sum_{0}^{\infty} a_i x^i$$

Let us consider the coding process over $\mathbf{F}_2$. Given a streams over $\mathbf{F}_2$, $(a_0, a_1, a_2, a_3, \cdots, a_n, \cdots)$, i.e., $a_i \in \mathbf{F}_2$. we may use the following process to produce two (in general, maybe 1, or any finitely many) data streams $(b_0, b_1, \cdots, b_n, \cdots), (c_0, c_1, \cdots, c_n, \cdots)$.



where at the time 0, a_0 enters the leftest square at the middle and the other two quares at the middle row are with 0's, the two $\oplus$ at the top and bottom denote module 2 additions. As time goes from n to $n+1$, all values in the middle boxes will move to their right, so the rightest one will fall off and disappear, the $n+1$th coefiecient will occupy the leftest box.

Mathematically, let us use

$$g_1(x) = \sum_0^\infty b_i x^i, \qquad g_2(x) = \sum_0^\infty c_i x^i$$

where

$$\begin{aligned} &b_0 = a_0,\ b_1 = a_1,\ b_n = a_n + a_{n-2} \text{ for all n>1} \\ &c_0 = a_0,\ c_1 = a_1 + a_0,\ b_n = a_n + a_{n-1} + a_{n-2} \text{ for all n>1} \end{aligned}$$

Or we simply write

$$\begin{aligned} g_1(x) &= (1+x^2)f(x) \\ g_2(x) &= (1+x+x^2)f(x) \end{aligned}$$

It seems that a encoder is simply a multiplication by a polynomials. However there is a catch: a multiplication by x (a delay of time) should be considered as invertible (at least left-invertible)! So we should enlarge the polynomial ring $\mathbf{F}[x]$ to the Laurent polynomial ring $\mathbf{F}[x]_{(x)} = \{\frac{h(x)}{x^d} : h(x) \in \mathbf{F}[\text{x}]\}$.

If we let the Laurent polynomial ring $\mathbf{F}[x]_{(x)}$ acts on the power series ring $\mathbf{F}[[x]]$, then we find it is not even closed under the action induced by x^{-1}. The natural things to do is to enlarge the power series ring $\mathbf{F}[[x]]$ to the meromorphic function field $\mathbf{F}((x))$. Recall that

$$\mathbf{F}_2((x)) = \{\sum_{-m}^\infty a_i x^i : a_i \in \mathbf{F}_2\}$$

In the above diagram, the encoded stream $g_1(x)$ is defined by $g_1(x) = (1+x^2)f(x)$. Suppose there is no error, can we recover $f(x)$ from $g_1(x)$? Mathemtically, $f(x) = (1+x^2)^{-1}g_1(x)$, however, there are two problems: (1): the recursive formula $a_0 = b_0, a_1 = b_1, a_2 = b_2 - b_1, a_3 = b_3 - b_2 + b_1, \cdots$ is getting longer and longer without a bound. Therefore, we require an unbounded number of memory units. (2). If we have a single error for $g_1(x)$, namely replacing $g_1(x)$ by $g_1(x)+x^n$, then the inverse will differ with $f(x)$ at infinitly many places. Any encoder with the second type problem will be called a *catastrophic encoder* and will not be used.

A.2 Combining and Splitting

It is clear that the only good encoders which take one incoming stream of data $f(x)$ and produce one stream of data $g_1(x)$ are multipling by x^n. Those are non-interesting. How about a encoder which produces more streams of data as in our preceeding diagram? Let us first study the technique of combining and splitting data streams. Let

$$f(x) = \sum_{i=-m}^{\infty} a_i x^i$$
$$g_j(x) = \sum_{i=-m}^{\infty} b_{ji} x^i \text{ for j=1,2,}\cdots\text{,r}$$

Then we have $h_j(x), h(x)$ uniquely defined by the following equations,

$$f(x) = \sum_{j=0}^{n-1} x^i h_j(x^n)$$
$$h(x) = \sum_{j=1}^{r} x^{j-1} g_j(x^r)$$

In other words, we may split one stream of data $f(x)$ into n streams of data $h_j(x)$ for $j = 1, 2, \cdots, n$, in symbol $S_n(f(x)) = [h_0(x), h_1(x), \cdots, h_{n-1}(x)]$, and combining r streams of data $g_j(x)$ for $j = 1, 2, \cdots, r$ into one stream of data $h(x)$, in symbol $C_r(g_1(x), \cdots, g_r(x)) = h(x)$. The splitting and combining operations are one to one and onto maps between $\mathbf{F}((x))$ and $\mathbf{F}((x))^r$ or $\mathbf{F}((x))^n$ while they are non-linear respect to the field $\mathbf{F}((x))$.

A.3 Smith normal form

Our preceding figure over $\mathbf{F}_2$ can be written mathematically as

$$[f(x)][1 + x^2, 1 + x + x^2] = [g_1(x), g_2(x)].$$

Using some linear algebra, we may rewritten the above matrix equations as

$$\begin{bmatrix} f(x) \end{bmatrix} \cdot \begin{bmatrix} 1 \end{bmatrix} \cdot \begin{bmatrix} 1, & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 + x^2, & 1 + x + x^2 \\ x, & 1 + x \end{bmatrix} = \begin{bmatrix} g_1(x), & g_2(x) \end{bmatrix}$$

In other words, we have the following matrix equation

$$\begin{bmatrix} 1 + x^2, 1 + x + x^2 \end{bmatrix} = \begin{bmatrix} 1 \end{bmatrix} \cdot \begin{bmatrix} 1, & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 + x^2, & 1 + x + x^2 \\ x, & 1 + x \end{bmatrix}$$

The above is the well-known *Smith normal form* of a matrix over a P.I.D. $\mathbf{R}$ as follows,

Proposition A.1 Let $\mathbf{R}$ be a P.I.D., and M an $r \times n$ matrix with entries in $\mathbf{R}$. Then M can be written as

$$M = A\Gamma B$$

where (1): both A and B are invertible such that their inverses are with entries in $\mathbf{R}$. (2) Γ is in the diagonal form with entries on the diagonal the invarient factors γ_i of M. Note that $\gamma_1 \mid \gamma_2 \mid \cdots \mid \gamma_r$.

Proof: Omitted. ■

In our present example, we have $A^{-1} = [1]$ and

$$B^{-1} = \begin{bmatrix} 1+x, & 1+x+x^2 \\ x, & 1+x^2 \end{bmatrix}$$

Therefore we have the following identity,

$$[g_1(x), g_2(x)] \cdot \begin{bmatrix} 1+x, & 1+x+x^2 \\ x, & 1+x^2 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \end{bmatrix} = [f(x)]$$

We conclude that with only finitely many memory units (in fact, at most 4), we may recover $f(x)$ if there is no error for $g_1(x)$ and $g_2(x)$. Furthermore, if there are single errors for $g_1(x)$ and $g_2(x)$, say replacing them by x^n, x^m, then the decoding result is a polynomial which is not an infinitely long meromorphic function. Therefore the encoder is not a catastrophic encoder.

In general, we may consider r streams of data $J = [f_1(x), \cdots, f_r(x)]$ where $f_i(x) \in \mathbf{F}((\mathrm{x}))$, we may view it as a vector J in the r-dimensional vector space $\mathbf{F}((\mathrm{x}))^r$. An encoder may be viewed as a $r \times n$ matrix M with coeffients in the Laurant polynomial ring $\mathbf{F}[\mathrm{x}]_{(x)}$ to produce n streams of data as a vector T in $\mathbf{F}((\mathrm{x}))^n$ in the following formula,

$$JM = T$$

According to the above Smith normal form, the above equation can be rewritten as

$$JA\Gamma B = T$$

Since both A, B are invertible, I is uniquely determined by $T \Longleftrightarrow \Gamma$ is invertible $\Longleftrightarrow$ invariant factors of M are invertible, i.e., they are powers of x. Then and only then we have a non-catastrophic encoder.

A.4 Viterbi Algorithm

There are several decoding methods. All are based on maximum likehood decoding and are not very difficulty. One is Viterbi algorithm using the *state diagrams*, the other is the sequential decoding (or Fano algorithm) using the *tree diagram*.

Among all possible decoding methods known today, the Viterbi Algorithm is the best. Let us discuss the *Viterbi algorithm*. We will define the *state* as the ordered tuple of bits in the memory units. For instance, look at the representation of § A.1, at the beginning, in the both memory units, one has $0, 0$, so it's *state* is 00, and at the box of split, one has a_0, let us assume it is 1. Then at the next step, the state becomes 10, and at the box of split, one has a_1. We shall view the shifting (or feeding the data) as a transfermation of *states*. The sequence of messages $[a_0 a_1 a_2 \cdots]$ uniquely determines the sequence of the corresponding states and vice versa.

Let us consider an example with $f(x) = 1 + x + x^4 + x^6$ as in the representation of § A.1, then $g_1(x) = 1 + x + x^2 + x^3 + x^4 + x^{16}$, $g_2(x) = 1 + x^3 + x^4 + x^5 + x^7 + x^8$ and $h(x) = g_1(x^2)+xg_2(x^2) = 1+x+x^2+x^4+x^6+x^7+x^8+x^9+x^{11}+x^{15}+x^{16}+x^{17}$. However a noisy channel produces a received word $r(x) = 1+x+x^2+x^6+x^8+x^9+x^{11}+x^{15}+x^{16}+x^{17} = [11, 10, 00, 10, 11, 01, 00, 01, 11]$. How do we recover the original $h(x)$? In other words, how do we recover the original sequence of states? We shall consider all possible transfermations of states, starting with the state 00, if the input bit is 1 (i.e., $a_0 = 1$), then the output is [11] and the next state will be 10, if the input bit is 0, then the output is [00] and the next state will be 00. If we consider only the possible sequences of states, then it is easy to see that after t steps, we will consider 2^t possible paths. The number is explosively large. We shall prune all possible sequences of states using the Hamming distance of the output sequence to the received word $r(x)$ as the criterion.

Let us consider the following **diagram 1 of paths** of $1 \le t \le 3$,

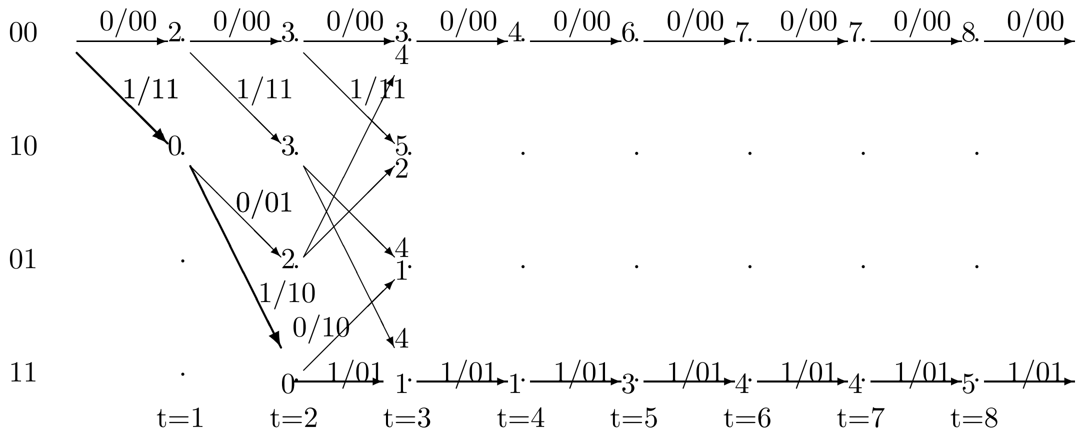


Diagram 1 of Paths

We write the *input/output* on most arrow, and write the Hamming distance of the output sequence and the truncated r at the tip of an arrow. For instance, for the path determined by a sequence of input [001], the output is [00, 00, 11], comparing with the received sequence $r = [11, 10, 00, \cdots]$, we find that the Hammong distance is 5. For the path DETermined by a sequence of input [110], the output is [11, 10, 10], comparing with the received sequence $r = [11, 10, 00, \cdots]$, we find that the Hammong distance is 1. Look at the above diagram up to $t = 3$, we realize that since the minimal distance is 1 between all allowed sequences and the received one, there must be some error(s). Furthermore, there are four pairs of paths where for the two paths in the same pair, each one starts and end at the same points. Comparing the two paths in each pair, we may delete the one with larger Hamming distance from the received r. This is the *pruning principle.* We shall use the pruning principle to keep only 4 paths for consideration at t (instead of 2^t possible paths) and extend the above diagram from $t = 3$ to $t = 8$. We have the following diagram with the details lefted to the reader. We will select the path with Hamming distance 2 to the received word r as the maximum likehood decoded result. From the selected path, we decide that the maximum likehood code word is

$[11, 10, 10, 11, 11, 01, 00, 01, 11]$ which happens to be the original code word.

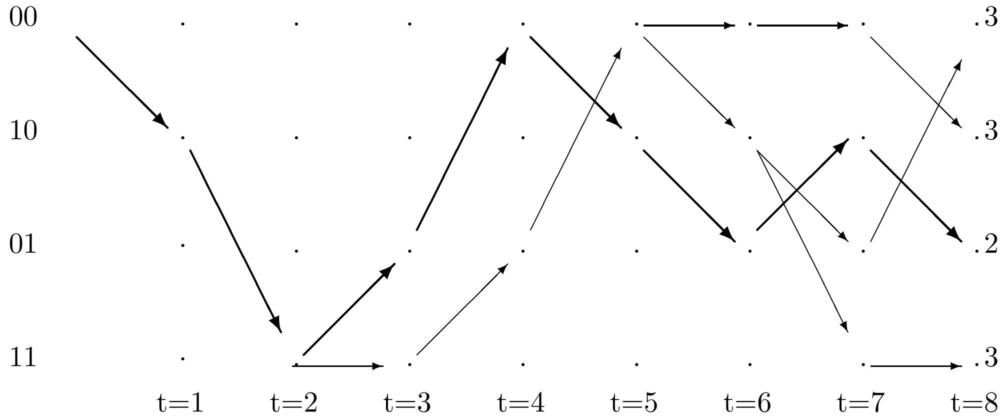
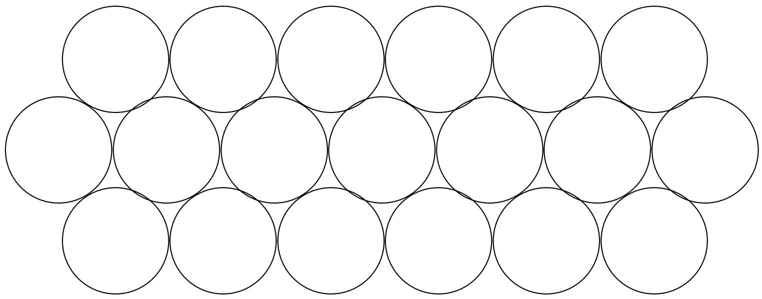


Diagram 2 of Paths

Appendix B

Sphere Packing Probelm and Weight Enumerators

To tile a real plane with same shape, same size tiles, one may use same sized triangles, equalaterials or hexagons. For all other shapes there are always empty spaces left. We may consider the problem of filling a plane with the same size discs with gaps allowed. One knows from experience that the following arrangement for discs is tight,



In 1611, J.Kepler conjectured that the best space-saving way to stack equal sized balls in $\mathbf{R}^3$ was already known to all fruit sellers in the market, the pyramidal arrangement. In the 1990's there are two controversial papers solving this 400 years old conjecture, one by W.Y.Hsiang, another by T.C.Hale. For the higher dimensional cases, the only known solutions are for dimensions 4, 8 and 24.

The relation of sphere packing problem and the self-correcting code is as follows. Let us take the plane example. If both the sender and the receiver select the centers of the discs as the permissible code words, and the receiver receives a point which is slightly different from the point of the original message. We believe that the distance between those two points indicate the measure of error occured. As long as the received point is in a disc (which is likely), the receiver will decode it as the center of the disc. Naturally, we want to pack the space by the most efficient way so we may have the largest possible spheres in some region of the plane. We shall generalized the sphere packing problem to a high dimension.

B.1 Golay Code

Let us consider the beautiful [23,12] Golay code. In the vector space $\mathbf{V}=\mathbf{F}_2^{23}$, a ball of radius 3 centered at $[00\cdots00]$ contains at least

$$\sum_{i=0}^{3} C_i^{23} = 2048 = 2^{11}$$

points. It is easy to see any ball of radius 3 contains exactly 2^{11} elements. In the vector space $\mathbf{V}$, there are 2^{23} elements, so it is possible to have 2^{12} balls with radius 3 and do not overlap. Indeed this happens (see below) and we will have a code with 2^{12} code words of length 23 and the information rate $12/23 = 0.52$.

We shall follow the way of constructing the Reed-Solomon code to construct Golay code. Let us consider the field $\mathbf{F}_{2^{11}}$. All non-zero elements in $\mathbf{F}_{2^{11}}$ satisfy the following equation

$$x^{2^{11}-1} + 1 = 0$$

Note that

$$2^{11} - 1 = 23 \times 89$$

Therefore, $\mathbf{F}_{2^{11}}$ contains a $23rd$ root of unity. We have the following decomposition,

$$x^{23}+1 = (x+1)(x^{11}+x^9+x^7+x^6+x^5+x+1)(x^{11}+x^{10}+x^6+x^5+x^4+x^2+1) = (x+1)g(x)g^*(x).$$

We have the following definition,

Definition : The [23,12] Golay code consists of all polynomials $h(x)g(x)$ in $\mathbf{F}_2[x]/(x^{23}-1)$ with deg(h(x))$\leq$ 11. ∎

Note that the [23,12] Golay code is not a Reed-Solomon code. We have the following Proposition,

Proposition B.1.1 : *The balls of radius* 3 *around the code words pack* $\mathbf{V}$.

Proof. : Omitted. ∎

The other Golay code is [11,6] Golay code over $\mathbf{F}_3$ where balls of radius 2 densely pack $\mathbf{F}_3^{11}$.

B.2 Uniformly packed code

The perfect codes over a finite field is like tiling of the plane. The only perfect codes are (1): repetition code of odd length. (2) q-ary Hamming codes (cf Exercise 1.5 (1), (2), (4)) and (3): [23,12] binary Golay code and [11,6] tenary Golay code. Otherwise, it is impossible to use balls of the same radius to tile a vector space $\mathbf{F}_q^n$. Let us consider the simple case over $\mathbf{F}_2$. We study

> $A(n,d) =$ the largest integer M such that there exist M codewords $\{x_1, \cdots, x_M\}$ such that $d(x_i, x_j) \geq d$, if i≠j.

Let d be odd. Then the number $A(n,d)$ is the maximal number of putting balls of radius $\lfloor d/2 \rfloor$ in $\mathbf{F}_2^n$ without touching each other. There might be some extra space allowed. This is similar to sphere packing problem in $\mathbf{R}^3$. We may form a code C with $C = \{x_1, \cdots, x_M\}$ which will correct $\lfloor d/2 \rfloor$ errors. In general, it is difficult to find the number $A(n,d)$ except some easy cases with n, d small or $d \geq n$.

We may put more conditions on the code C to make it easier to construct. Let us define the distance $d(x, C)$ and the covering radius $\rho(C)$ of a set C as

$$d(x, C) = min\{d(x, c) : c \in C\}$$
$$\rho(C) = max\{d(x, C) : x \in \mathbf{F}_2^n\}$$

We have the following definition,

Definition : Given a code C which may not be linear. Let e be a positive integer such that the minimal distance d of C satisfies $d \geq 2e+1$ and $\rho(C) = e+1$. Then C is said to be an uniformly packed code with parameter r if every word x with $d(x, C) \geq e$ has distance d or $d+1$ to exactly r code words. ■

Let us consider the condition $d(x, C) \geq e$. If $d(x, C) = e$, then there is an unique $c \in C$ such that $d(x, c) = e$. so we conclude that

$$r - 1 \leq \frac{n-e}{e+1}$$
$$r \leq \frac{n+1}{e+1}$$

If $d(x, C) = e+1$, then we have

$$r \leq \frac{n}{e+1}$$

We define a *nearly perfect code* as an uniformly packed code with parameter $r = \lfloor \frac{n}{e+1} \rfloor$. After years reseach, it turns out that there are few uniformly packed codes and nearly perfect codes (cf page 126, [7]).

B.3 Weight Enumerators

For any code C, it is meaningful to compute the geometric configuration $\{d(c_i, c_j) : c_i, c_j \in C\}$ of C defined by the Hamming distance. In general the computation is complicated. We may restrict C to linear $[n, k]$ code only. Then we may take $c_i = 0$ and define $A_\ell = | \{c_j : c_j \in C,\ wt(c_j) = \ell\} |$. As a standard trick in mathematics, we form the following polynomial, which is called the *weight enumerator* of C,

$$A(z) = A_0 + A_1 z + \cdots + A_n z^n$$

We have clearly $A_0 = 1$ and $A(1) = q^k$. we have the following example;

Example B1: Let C be the repeated code with generating matrix [11111]. Then $C = \{[00000], [11111]\}$ and $A(z) = 1 + Z^5$. ■

However, to compute $A(z)$ for large q, we still have to look over all q^k vectors in C. It is still a formidable task. If $n - k$ is small, then we may be able to compute the weight enimerator for the dual code $C^\perp$. The following Propersition is due to J.MacWilliams,

Proposition B.3.1 *(MacWilliams identity): Let $A(z)$ be the weight enumerator of an [n,k] linear code C and $B(z)$ be the weight enumerator of the dual code $C^\perp$. Then $A(z)$ and $B(z)$ are related by the following formula,*

$$B(z) = \frac{1}{q^k} \sum_{i=0}^{n} A_i (1 - z)^i (1 + (q - 1)z)^{n-i}$$

Proof. : The reader is referred to (page 148, [4]) or (page 41, [7]). ■

Example B2: Let $C = [7, 4]$ Hamming code. Then its dual code $C^\perp$ is an $[7, 3]$ code. It turn out each non-zero code words in $C^\perp$ has weight 4. Thus the weight enumerator of $C^\perp$ is $1 + 7z^4$. Therefore the weight enumerator of C is

$$\frac{1}{8}[(1 + z)7 + 7(1 - z)^4(1 + z)^3] = 1 + 7z^3 + 7z^4 + z^7$$

■

Appendix C

Other Important Coding and Decoding Methods

C.1 Hadamand Codes

The space ship US Mariner 1969 carried a *Hadamand code*. We will introduce the *Hadamand matrix* which is defined as follows,

Definition : A square matrix H_n of order n with entries $+1$ or -1 such that $H_n H_n^T = nI$ is called a *Hadamand matrix*. ■

Example C 1: We give the following examples,

$$H_2 = \begin{bmatrix} 1, & 1 \\ 1, & -1 \end{bmatrix} = \begin{bmatrix} +, & + \\ +, & - \end{bmatrix}$$

$$H_4 = \begin{bmatrix} 1, & 1, & 1, & 1 \\ 1, & 1, & -1, & -1 \\ 1, & -1, & 1, & -1 \\ 1, & -1, & -1, & 1 \end{bmatrix} = \begin{bmatrix} +, & +, & +, & + \\ +, & +, & -, & - \\ +, & -, & +, & - \\ +, & -, & -, & + \end{bmatrix}$$

■

It can be shown that there are *Hadamand matrices* only of order 2 or $4m$. In general given any integer $n = 4m$, we do not know if there is a *Hadamand matrix* of order n. The smallest unknown case is $n = 188$. In H_n and $-H_n$ we replace -1 by 0. Thus we have $2n$ rows which are codewords in $\mathbf{F_2}^n$. It is easy to see that all codewords are of equidistance $\frac{n}{2}$. Thus we have a code, a *Hadamand code*, [n,2n,n/2]. If n is not a power of 2, then the *Hadamand code* is nonlinear. The best codes with $n \leq 2d$ are practically all nonlinear and related to *Hadamand matrix*.

C.2 Reed-Muller Code

There are several ways to present the Reed-Muller code which is a binary code correcting several errors. The code was discovered by Muller in 1954 ([26]) and the decoding method was due to Reed in 1954 [28]. The decoding method of it is easy, hence it has been used in several occasions as that from 1969-1977 all of NASA's (USA) *Mariner* class deep-space probes were equipped with a Reed-Muller code (i.e., $RM(5,1)$, see below for definition).

Let $P(m,r)$ denote the set of all polynomials of degree $\leq r \leq m$ in m variables $(x_1, \cdots, x_m)$ over $\mathbf{F}_2$. Note that computationwise, over $\mathbf{F}_2$, we always have

$$x^2 = x$$

Hence all monomials can be reduced to mutiples of distinct x_i. Let $n = 2^m$ and $(v_0, v_1, \cdots, v_{n-1})$ denote a list of all the 2^m binary vectors in some order. Then for each $f \in P(m,r)$, we may define $f(v_j) = f(a_{j1}, \cdots, a_{jm})$ where $v_j = [a_{j1} \cdots a_{jm}]$, furthermore we get a binary vector of length 2^m via the mapping $f \mapsto (f(v_0), f(v_1), \cdots, f(v_{n-1}))$. The set of all vectors obtained in this way is called the *rth order Reed-Muller code* of length 2^m, or $RM(m,r)$ for short. It is easy to see that $RM(m,r)$ is a linear code $[n,k,d]$ (in fact it is a cyclic code). We will find the parameters k, d.

Proposition C.2.1 *: We have*

$$k = 1 + C_1^m + \cdots + C_r^m$$

Proof. The number of monomials which are mutiples of distinct x_i is clearly

$$1 + C_1^m + \cdots + C_r^m.$$

Therefore we need only to show that they are independent. Suppose that $f(x_1, \cdots, x_m)$ is a linear combination of those monomials, and $f(v_i) = 0$ for all i. If $f(0, x_2, \cdots, x_m) \neq 0$, then we can find values for $x_2, \cdots, x_m$ so that it is not zero by mathematical induction. If $f(0, x_2, \cdots, x_m) = 0$, then $f(x_1, \cdots, x_m) = x_1 g(x_2, \cdots, x_m)$ and $f(1, x_2, \cdots, x_m) = g(x_2, \cdots, x_m)$, so that we can find values for $x_2, \cdots, x_m$ such that g is not zero by mathematical induction. In any case, we have a contradiction. ∎

Corollary C.2.2 *The vector space of all polynomials in m variables of degree less than or equal to r and having all terms with degree in each x_i is one or zero is isomorphic to $RM(m,r)$.*

Proposition C.2.3 *: We have the minimal distance $d = 2^{m-r}$.*

Proof. We first show that $d \leq 2^{m-r}$. Let $f = x_1 x_2 \cdots x_r$. Then $f = 0$ whenever $x_1 = 0$ or $x_2 = 0$ or $\cdots$ or $x_r = 0$. Using set-theoretic inclusions to compute the number of roots, we may conclude that it is $2^m - 2^{m-r}$. There are at least $2^m - 2^{m-r}$ zeroes among $v_0, \cdots, v_{n-1}$. Therfore the minimal weight, hence the minimal distance, is at most 2^{m-r}.

We wish to prove that $d \geq 2^{m-r}$. Let us consider a polynomial $f(x_1, x_2, \cdots, x_m) \neq 0$ so every variable appears at most once in every term. We have $f(0, x_2, \cdots, x_m) = 0 \Longrightarrow f(x_1, \cdots, x_m) = x_1 g(x_2, \cdots, x_m)$. Similarly, $f(1, x_2, \cdots, x_m) = 0 \Longrightarrow f(x_1, \cdots, x_m) = (x_1 - 1)g(x_2, \cdots, x_m)$. After we try every x_i, then we conclude that either that there is an x_i, say x_1, such that $f(0, x_2, \cdots, x_m) \neq 0$ and $f(1, x_2, \cdots, x_m) \neq 0$ or $f = \prod_{i=1}^{m}(x_i - \delta_1)$ where $\delta_i = 0, 1$. The second case happens only if $r = m$, and $2^{m-r} = 1$, our proposition is certainly true. In the first case, both $f(0, x_2, \cdots, x_m) = g(x_2, \cdots, x_m) \neq 0$ and $f(1, x_2, \cdots, x_m) = h(x_2, \cdots, x_m) \neq 0$. By induction on the number of variables, we conclude that $g(x_2, \cdots, x_m)$ and $h(x_2, \cdots, x_m)$ have at most $2^{m-1} - 2^{m-1-r}$ zeroes. Therefore f has at most $(2^{m-1} - 2^{m-1-r}) + (2^{m-1} - 2^{m-1-r}) = 2^m - 2^{m-r}$ zeroes. So the minimal weight, hence the minimal distance, is at least 2^{m-r}.

■

Proposition C.2.4 : *We have* $RM(m, r)^{\perp} = RM(m, m - r - 1)$.

Proof. Let us first compute the dimensions,

$$\begin{aligned} dim(RM(m,r)^{\perp}) &= 2^m - (1 + C_1^m + \cdots + C_r^m) \\ &= C_{r+1}^m + \cdots + C_{m-1}^m + 1 \\ &= C_{m-r-1}^m + \cdots + C_1^m + 1 \\ &= dim(RM(m, m-r-1)) \end{aligned}$$

Now we only need to show that every nomomial in $RM(m, m - r - 1)$ is in $RM(m, r)^{\perp}$, i.e., with $f = (\prod x_i)(\prod_j x_j)$, where $(\prod x_i) \in RM(m, m - r - 1)$ and $(\prod_j x_j) \in RM(m, r)$, the following sum is zero always, (note that after reducing the total degree of f by $x_i^2 = x_i$, we may take f to be a product of less than or equal to $m - 1$ distinct variables.)

$$\sum_{i=0}^{n-1}(f(v_i)) = 2^{m-deg(f)} = 0 \ mod\ 2$$

The above equation is obvious.

■

Example C1: Let us consider $m = 4$. Then we have the following table for the values of polynomials 1, x_1, x_2, x_3, x_4;

	0,	1,	2,	3,	4,	5,	6,	7,	8,	9,	10,	11,	12,	13,	14,	15
1	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1,	1
x_1	0,	1,	0,	1,	0,	1,	0,	1,	0,	1,	0,	1,	0,	1,	0,	1
x_2	0,	0,	1,	1,	0,	0,	1,	1,	0,	0,	1,	1,	0,	0,	1,	1
x_3	0,	0,	0,	0,	1,	1,	1,	1,	0,	0,	0,	0,	1,	1,	1,	1
x_4	0,	0,	0,	0,	0,	0,	0,	0,	1,	1,	1,	1,	1,	1,	1,	1

Table C 1

For the rest of monomials, we use the formula $fg(v_j) = f(v_j)g(v_j)$ to find their values. In this way we nay construct a table for the values of all monomials. ■

The method of decoding RM codes is the *threshold decoding* in the section 4. The reader is referred to [28].

C.3 Constructing New Code from Old Codes

Extend, Puncture, Shorten, Lengthen, Expurgate, and Augment a Code

Let us take an example to explain the meanings of the operations. Let C_i be a linear code with the following generator matrix G_i and check matrix H_i. Let G_0, H_0 be given as follows. Note that C_0 is a cyclic code with generating polynomial $g_0(x) = x^3 + x + 1$.

$$G_0 = \begin{bmatrix} 1, & 0, & 0, & 0, & 1, & 1, & 0 \\ 0, & 1, & 0, & 0, & 0, & 1, & 1 \\ 0, & 0, & 1, & 0, & 1, & 1, & 1 \\ 0, & 0, & 0, & 1, & 0, & 1, & 1 \end{bmatrix}, \quad H_0^T = \begin{bmatrix} 1, & 0, & 1, & 0, & 1, & 0, & 0 \\ 1, & 1, & 1, & 1, & 0, & 1, & 0 \\ 0, & 1, & 1, & 1, & 0, & 0, & 1 \end{bmatrix}$$

Extend: Any code can be extended by annexing additional check symbols. The resulting code is not cyclic in general. We have the following example $C_0 \to C_2$.

$$G_2 = \begin{bmatrix} 1, & 0, & 1, & 1, & 1, & 0, & 0, & 0 \\ 0, & 1, & 0, & 1, & 1, & 1, & 0, & 0 \\ 0, & 0, & 1, & 0, & 1, & 1, & 1, & 0 \\ 1, & 1, & 1, & 1, & 1, & 1, & 1, & 1 \end{bmatrix}, \quad H_2^T = \begin{bmatrix} 1, & 0, & 1, & 0, & 1, & 0, & 0, & 0 \\ 1, & 1, & 1, & 1, & 0, & 1, & 0, & 0 \\ 0, & 1, & 1, & 1, & 0, & 0, & 1, & 0 \\ 1, & 1, & 1, & 1, & 1, & 1, & 1, & 1 \end{bmatrix}$$

Puncture: Any code can be punctured by deleting some of its check symbols. We have an example of $C_2 \to C_0$. **Expurgating**: A cyclic code generated by the polynomial $g(x)$ can be expurgated to form another cyclic code by multiplying any additional factor into the generated polynomial. The most common expurgate code is the one generated by $g(x)(x-1)$. We have the following example $C_0 \to C_1$ with C_1 a cyclic code generated by $g_1(x) = (x-1)g_0(x)$.

$$G_1 = \begin{bmatrix} 1, & 0, & 1, & 1, & 1, & 0, & 0 \\ 0, & 1, & 0, & 1, & 1, & 1, & 0 \\ 0, & 0, & 1, & 0, & 1, & 1, & 1 \end{bmatrix}, \quad H_1^T = \begin{bmatrix} 1, & 0, & 1, & 0, & 1, & 0, & 0 \\ 1, & 1, & 1, & 1, & 0, & 1, & 0 \\ 0, & 1, & 1, & 1, & 0, & 0, & 1 \\ 1, & 1, & 1, & 1, & 1, & 1, & 1 \end{bmatrix}$$

Augment: A cyclic code generated by the polynomial $g(x)$ may be augmented into another code cyclic code of the same length whose generator polynomial is a factor of $g(x)$. The most common augmented code is the one generated by $g(x)/(x-1)$ (we assume that it is allowed, i.e., $g(1) = 0$). We have the example $C_1 \to C_0$ with C_0 a cyclic code generated by $g_0(x) = g_1(x)/(x-1)$. **Lengthen**: If the generator polynomial of an original binary cyclic code of length n contains the factor $(x+1)$, then this cyclic code can be lengthened to a linear code of length $n+1$ by annexing the vector $[11\cdots 1]$ to the code's generator matrix. We have the example $C_1 \to C_2$. **Shorten**: Any code can shortened by deleting message symbols. We have the example $C_2 \to C_1$ with C_1 a cyclic code.

To summerize the preceding discussion, we have the following figure.

$$\begin{tikzcd}[column sep=small]
 & \boxed{\mathrm{H}_0^T = \begin{bmatrix} 1, & 0, & 1, & 0, & 1, & 0, & 0 \\ 1, & 1, & 1, & 1, & 0, & 1, & 0 \\ 0, & 1, & 1, & 1, & 0, & 0, & 1 \end{bmatrix}} \arrow[dl, shift left, "\text{Expurgate}"] \arrow[dr, shift right, "\text{Extend}"'] & \\
\boxed{\mathrm{H}_1^T = \begin{bmatrix} 1, & 0, & 1, & 0, & 1, & 0, & 0 \\ 1, & 1, & 1, & 1, & 0, & 1, & 0 \\ 0, & 1, & 1, & 1, & 0, & 0, & 1 \\ 1, & 1, & 1, & 1, & 1, & 1, & 1 \end{bmatrix}} \arrow[ur, shift left, "\text{Augment}"] \arrow[rr, shift left, "\text{Lengthen}"] & & \boxed{\mathrm{H}_2^T = \begin{bmatrix} 1, & 0, & 1, & 0, & 1, & 0, & 0, & 0 \\ 1, & 1, & 1, & 1, & 0, & 1, & 0, & 0 \\ 0, & 1, & 1, & 1, & 0, & 0, & 1, & 0 \\ 1, & 1, & 1, & 1, & 1, & 1, & 1, & 1 \end{bmatrix}} \arrow[ul, shift right, "\text{Puncture}"'] \arrow[ll, shift left, "\text{Shorten}"]
\end{tikzcd}$$

Table C 2

Direct Product of Cyclic Codes

Let C_1, C_2 be two cyclic codes with block lengths n_1, n_2 and generated by polynomials $g_1(x), g_2(x)$ respectively. Let us assume that n_1, n_2 are co-prime, i.e., there are p_1, p_2 such that

$$n_1 p_1 + n_2 p_2 = 1 \ mod \ n_1 n_2.$$

Let use the *Chinese Remainder Theorem* to write

$$\ell = i \ mod \ n_1, \qquad \ell = j \ mod \ n_2$$

for all $0 \le \ell < n_1 n_2$. Let $b_0, b_1, \cdots, b_{n_1 n_2 - 1}$ be a block of length $n_1 n_2$. Then we may determine $a_{ij} = b_\ell$ according to the above formula. For instance, for $n_1 = 3, n_2 = 7$, we have the following matrices

$$\begin{bmatrix} a_{00}, & a_{01}, & a_{02}, & a_{03}, & a_{04}, & a_{05}, & a_{06} \\ a_{10}, & a_{11}, & a_{12}, & a_{13}, & a_{14}, & a_{15}, & a_{16} \\ a_{20}, & a_{21}, & a_{22}, & a_{23}, & a_{24}, & a_{25}, & a_{26} \end{bmatrix} = \begin{bmatrix} b_0, & b_{15}, & b_9, & b_3, & b_{18}, & b_{12}, & b_6 \\ b_7, & b_1, & b_{16}, & b_{10}, & b_4, & b_{19}, & b_{13} \\ b_{14}, & b_8, & b_2, & b_{17}, & b_{11}, & b_5, & b_{20} \end{bmatrix}$$

The ordering of $[a_{ij}]$ is called the *cyclic ordering* . Then we have the following Proposition,

Definition Let C_1, C_2 be two linear codes (not nessary cyclic) of lengths n_1, n_2 respectively. then the direct product code $C_1 \times C_2$ is the two dimensional code consisting of all codewords which satisfy the following conditions

1. For all j, $[a_{0j}a_{1j}\cdots a_{n_1-1,j}] \in C_1$,
2. For all i, $[a_{i0}a_{i1}\cdots a_{i,n_1-1}] \in C_2$

■

Proposition C.3.1 *(Burton-Weldon) [16] [4] With assumption as above, the cyclic ordering of the product code $C_1 \times C_2$ is a cyclic code of length $n_1 n_2$ generated by*

$$g(x) = GCD(g_1(x^{n_2 p_2}), g_2(x^{n_1 p_1}, x^{n_1 n_2} - 1))$$

Proof. See [16] ■

The product code can be used for burst of errors. Suppose we decode the rows first. There may be several rows which are *erasures* beyond correcting. Then we decode the columns. The rows which are completely erased and can not be recovered by decoding rows may be recovered by decoding columns. The codes used in the modern compact discs are product codes.

Cancatenated Codes and Justensen Codes

The concept of *concatenated codes* was introduced by G.D.Forney [20] in 1966. Let us consider a simple example. Suppose taht we will send out an e-mail which consists of several sentences in certain set of alphabets. We may first encode the alphabets in certain way. Say we have three alphabets, we may encode them in $\mathbf{F}_{2^3}$ as

$$\begin{array}{c} 00000 \\ 11100 \\ 00111 \end{array}$$

Note that the minimal distance of the three alphabets is 3, and we expect to correct 1 error. Then we may use some encoding scheme, say RS-code, to encode the whole e-mail. The way to encode the alphabets will be called the *inner code* and the way to encode the e-mail will be called the *outer code*. In general, we may encode any message by a code C_1 (like the encoding of all alphabets) which is called the *inner code*, and then encode the codewords by C_1 by another code C_2 which is called the *outer code*. The most important development along this direction is the *Justensen codes* [25].

C.4 Threshold Decoding

We may use the *threshold decoding* to handle the problem of decoding several codes. We will illustrate the threshold decoding by the following example.

Example C 2: Let C a $[7,3]$ binary linear code with the check matrix H,

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

Let $c = [c_0, c_1, \cdots, c_6]$ be the code word sent, and $r = [r_0, r_1, \cdots, r_6] = c + e$ be the received word where $e = [e_0, e_1, \cdots, e_6]$ is the error vector. This code may correct 1 error. We shall consider the code in a different way. Analytically, the check matrix produce the following system of equations satisfied by c_0,

$$\begin{aligned} c_0 &= c_1 + c_3 \\ c_0 &= c_2 + c_4 \\ c_0 &= c_5 + c_6 \end{aligned}$$

Now let us assume the channel is *binary symmetric* with error probability $\wp < \frac{1}{2}$. Then we have

$$c_0 = \begin{cases} r_1 + r_3 & : \quad \text{if } e_1 + e_3 = 0 \\ r_2 + r_4 & : \quad \text{if } e_2 + e_4 = 0 \\ r_5 + r_6 & : \quad \text{if } e_5 + e_6 = 0 \\ r_0 & : \quad \text{if } e_0 = 0 \end{cases}$$

If there is one error, then we may use *majority vote* to decide which value is correct for c_0. If two errors are allowed, and there is a $2-2$ tie on the vote, since $e_0 = 0$ is with a probability $1-\wp$ and $e_i + e_j = 0$ is with a probability $(1-\wp)^2 + \wp^2 = 1 - \wp - \wp(1-2\wp) < 1 - \wp$, so it is in favor of the value of r_0.

■

Reed obtained a generalization of the above technique to *RM* codes (cf [28]) which we will illustrate by the following example. Later on the method was applied to many codes and named *threshold decoding*. The interested reader is referred to MacWilliams and Sloane [5].

Example C 3: Let us consider $RM(3,1)$ which is a $[8,4,4]-$ code. We have the following table,

	0,	1,	2,	3,	4,	5,	6,	7,
1	1,	1,	1,	1,	1,	1,	1,	1,
x_1	0,	1,	0,	1,	0,	1,	0,	1,
x_2	0,	0,	1,	1,	0,	0,	1,	1,
x_3	0,	0,	0,	0,	1,	1,	1,	1,

Table C 2

We have the following method of decoding.

Threshold Decoding: Let the original message be $[m_0, m_1, m_2, m_3]$, the code word be $c = [c_0, c_1, \cdots, c_7] = m_0[1, 1, \cdots, 1] + m_1[0, 1, 0, 1, 0, 1, 0, 1] + m_2[0, 0, 1, 1, 0, 0, 1, 1] + m_3[0, 0, 0, 0, 1, 1, 1, 1]$ and the received word be $r = [r_0, r_1, \cdots, r_7]$. Then the relations between m_i, c_j are

$$c_0 = m_0, c_1 = m_0 + m_1, c_2 = m_0 + m_2, c_3 = m_0 + m_1 + m_2, c_4 = m_0 + m_3$$
$$c_5 = m_0 + m_1 + m_3, c_6 = m_0 + m_2 + m_3, c_7 = m_0 + m_1 + m_2 + m_3$$

The above relations can be written as follows,

$$m_1 = c_0 + c_1 = c_2 + c_3 = c_4 + c_5 = c_6 + c_7$$
$$m_2 = c_0 + c_2 = c_1 + c_3 = c_4 + c_6 = c_5 + c_7$$
$$m_3 = c_0 + c_4 = c_1 + c_5 = c_2 + c_6 = c_3 + c_7$$

Clearly we do not know c_i's and only know r_i's. Assume that there is at most 1 error. Let us consider the value of m_1, the values of m_2, m_3 can be considered similarly. Then at least 3 of the four values of $r_0 + r_1, r_2 + r_3, r_4 + r_5, r_6 + r_7$ must be correct. Therefore a majority vote will decide the correct value of m_1. If there are two errors, then the vote might be tied. In that case, the value is indetermined. After we find the values for m_1, m_2, m_3, we are left to find m_0. Note that

$$[c_0, c_1, \cdots, c_7] = m_0[1, 1, \cdots, 1] + m_1[0, 1, \cdots, 0, 1] + m_2[0, 0, 1, \cdots, 1, 1] + m_3[0, 0, \cdots, 1, 1]$$

Once we find the correct values of m_1, m_2, m_3, let us compute

$$[r_0, r_1, \cdots, r_7] + m_1[0, 1, \cdots, 0, 1] + m_2[0, 0, 1, \cdots, 1, 1] + m_3[0, 0, \cdots, 1, 1]$$

The result of the above sum should be close to either $[1, 1, \cdots, 1]$ or $[0, 0, \cdots, 0]$. A majority vote should give us the correct value of m_0 (except the case of a $4 - 4$ tie, then the value of m_0 is indetermined).

■

C.5 Soft-Decision Decoding

The practics of transmitting signals electronically involve wave functions through a medium. After a white noise is added to the wave function, there is a distortion to the signals. Usually, one can not decide surely if the signal s is 0 or 1 (we only consider binary bits). One can associate reliability probabilities p_0 and p_1 (note that $p_0 + p_1 = 1$) to the particular signal s being 0 and 1. We usually decide the bit is 0 if $p_0 \geq 1/2$ and 1 if $p_0 \leq 1/2$. We consider that if p_0 is close to 1, then it is reliable to assign the signal to 0, and if it is close to 0, then it is reliable to assign the signal to 1. If it is close to 1/2, then the reliability is weak. One may use the values of p_0 and p_1 to decode the sequeence of signals. This is what called *soft-decision* decoding. On the other hand, if we only use the bits, without considering the reliability probabilities of them, to decode then it is called *hard-decision* decoding. So far in this book we only consider hard-decision decoding.

Let us cosider that the original messege is $c = (c_0 c_1 \cdots c_n)$ and the received word is $r = (r_0 r_1 \cdots r_n)$ with the reliabilities sequeence $(p_0^{(1)} \cdots p_0^{(n)})$, the Forney's *generalized minimum distance* (GMD) decoder [20] is essentially to erase $d-1$ least reliable symbols, and then use an algebraic decoder to correct the remaining received word.

A more interesting method is to use the reliability probabilities $(p_0^{(1)} \cdots p_0^{(n)})$. Let us consider a 3 repetition code which for certain bit produces the reliability probabilities $(0.51, 0.51, 0.12)$ for $(0, 0, 0)$. A hard-decision (which we had considered so far) decoder will first make a decision that the bit received are $(0, 0, 1)$ (since $0.51 > 0.50, 0.12 < 0.50$) and then use the majority rule to decide the original bit is 0. A soft-decision decoder will take the reliability probabilities into consideration. It would take the average reliability probability as the deciding factor; the average reliability probability is

$$\frac{1}{3}(0.51 + 0.51 + 0.12) = 0.38$$

Therefore, the original bit is more likely to be 1. In general, it can be shown that the soft-decision decoder is better.

C.6 Turbo Decoder

iIn 1993, a pair of French engineers, Claude Berrou and Alain Glavioux, announced at IEEE International Conference on Communications in Geneva their *Turbo Codes* [15] which come very close to Shannon's theoretical results. It was a surprising news to the world of coding theory.

Berrou likens the code to a crossword puzzle in which one would-be solver (decoder) receives the "across" clues and the other receives the "down" clues. After the decoders update their proposed solutions, they compare notes and update their solutons again. They continue this process either a limited number of times (say, 18 times) or they reach a consensus about the original message.

C.7 Low-Density Parity Check (lDPC) Codes

There is another code which rivals the Turbo codes, the Low-Density Parity Check (LDPC) Codes [21] which was created by Robert Gallager, who was a Ph. D. student at MIT, in 1958. This code is sometimes called Gallager code. First, Gallager used *sparse matrices* for the generator matrices. Second, Gallager used a decoder for every bit, and let the decoders talk among them. Thus created a huge rumor mill with thousands or tens of thousands talkers. The patent right of LDPC was held by Codex Corp until it expired without ever being used, One of the reasons was it was techniquely infeasible to create the rumer mill in the 50'.

Appendix D

Berlekamp's Decoding Algorithm

Recall that in § 3.2 we define the error-locator polynomial $\sigma(x)$ and the error-evaluator polynomial $\omega(x)$ as

$$\begin{aligned}\sigma(x) &= \prod_{i\in M}(1-\gamma^i x) = 1+\cdots\\ \omega(x) &= \sum_{i\in M} e_i\gamma^i x \prod_{j\in M\setminus i}(1-\gamma^j x)\end{aligned}$$

Let

$$\begin{aligned}r(x) &= r_0 + r_1 x + \cdots + r_{n-1}x^{n-1} = \text{the received word}\\ S(x) &= \sum_{i=1}^{2t} r(\gamma^i)x^i\end{aligned}$$

Then we have the following equation,

$$S(x)\sigma(x) = \omega(x)\ mod\ x^{2t+1}.$$

We may slightly modify the above equation as

$$(1+S(x))\sigma(x) = \sigma(x)+\omega(x)\ mod\ x^{2t+1}.$$

We may abuse the notations to write $\sigma(x)+\omega(x)$ as $\omega(x)$ (for the sake of induction later on), and get the following *key equation*,

$$(1+S(x))\sigma(x) = \omega(x)\ mod\ x^{2t+1}.$$

A central problem in coding theory is given $S(x)$, how do we fastly produce $\sigma(x)$ and $\omega(x)$? One of the simple methods is to look at the conditions and consider

$$\begin{aligned}S(x) &= s_1 x + s_2 x^2 + \cdots + s_{2t}x^{2t}\\ \sigma(x) &= 1 + a_1 x + a_2 x^2 + \cdots + a_t x^t\end{aligned}$$

where s_i are known and a_i are unknown. Since deg($\bar{\omega}(x)$)$\leq t$, we will set the coefficients of $x^{t+1}, \cdots, x^{2t}$ in the product of $S(x)\sigma(x)$ to be zeroes, and get the following linear equations,

$$
\begin{aligned}
&s_t a_1 + s_{t-1}a_2 + \cdots + s_1 a_t + s_{t+1} = 0\\
&s_{t+1}a_1 + s_t a_2 + \cdots + s_2 a_t + s_{t+2} = 0\\
&\cdots\cdots\cdots\\
&s_{2t-1}a_1 + s_{2t-2}a_2 + \cdots + s_t a_t + s_{2t} = 0
\end{aligned}
$$

Since we know that the above system of equations is consistent and produces an unique solution for indeterminant $a_1, \cdots, a_t$. We just solve it, find $\bar{\sigma}(x)$ and take $\omega(x) = S(x)\sigma(x)\ mod\ x^{2t+1}$. However, this method of solving system of equations is slow. In this section we will reproduce the original method of Berlekamp (1968 [2]) which is still the fastest method today.

Berlekamp Algorithm

Note that we already know the existence of $\sigma(x)$ and $\bar{\omega(x)}$, it follows from Proposition 2.7.5 that they are unique. The only problem is how to find them fast. Note that we know deg($\sigma(x)$)$\leq t$, deg($\bar{\omega}(x)$)$\leq t$ and deg(S(x))$\leq 2t$. Berlekamp's idea is not to solve the above system of linear equations directly, rather to find a sequence of $\sigma^{(k)}(x)$ and $\bar{\omega}^{(k)}(x)$ with $\sigma^{(2t)}(x) = \sigma(x)$ and $\bar{\omega}^{(2t)}(x) = \bar{\omega}(x)$, the *key equation* is generalized to a sequence of equations of the following form,

$$S(x)\sigma^{(k)}(x) = \bar{\omega}^{(k)}(x)\ mod\ x^{k+1} \qquad (D1_k)$$

with some degree restrictions on the polynonials $\sigma^{(k)}(x), i\bar{\omega}^{(k)}(x)$. Inductively, suppose that we have constructed $\sigma^{(k)}(x), \bar{\omega}^{(k)}(x)$, let us look at one more term as follows

$$S(x)\sigma^{(k)}(x) = \bar{\omega}^{(k)}(x) + \Delta^{(k)}x^{k+1}\ mod\ x^{k+2}$$

where $\Delta^{(k)}$ is a number. If $\Delta^{(k)} = 0$, then we may take $\sigma^{(k+1)} = \sigma^{(k)}$ and $\bar{\omega}^{(k+1)}(x) = \bar{\omega}^{(k)}(x)$ and continue our inductive process of construction. Supose $\Delta^{(k)} \neq 0$. Then we have to do something more. Consider

$$
\begin{aligned}
\Delta(\sigma^{(k)}) &= \sigma^{(k)} - \sigma^{(k+1)} = \Delta^{(k)}x\tau^{(k)}\\
\Delta(\bar{\omega}^{(k)}) &= \bar{\omega}^{(k)} - i\bar{\omega}^{(k+1)} = \Delta^{(k)}x\gamma^{(k)}
\end{aligned}
$$

Then we deduce the following equation,

$$S(x)\tau^{(k)} = \gamma^{(k)} + x^k\ mod\ x^{k+1} \qquad (D2_k)$$

Inductively, we have to define $\tau^{(k+1)}$ and $\gamma^{(k+1)}$. We may use one of the following two ways, if $\Delta^{(k)} = 0$,

$$\tau^{(k+1)} = x\tau^{(k)}, \text{ and } \gamma^{(k+1)} = x\gamma^{(k)} \qquad (D3)$$

or if $\Delta^{(k)} \neq 0$,

$$\tau^{(k+1)} = \frac{\sigma^{(k)}}{\Delta^{(k)}}, \text{ and } \gamma^{(k+1)} = \frac{\omega^{(k)}}{\Delta^{(k)}} \qquad (D4)$$

The critical things are to control the degrees of $\sigma^{(k)}(x), \bar{\omega}^{(k)}(x)$. We wish that they are less than or equal to $\frac{k+1}{2}$.

Initially, Berlekamp defines $\sigma^{(0)} = 1, \omega^{(0)} = 1, \tau^{(0)} = 1, \gamma^{(0)} = 0$, and integer valued functions $D(0) = 0$, $B(0) = 0$. Inductively, we have two cases:

$$\text{Case 1} = \begin{cases} \Delta^{(k)} = 0 & \text{or} \\ \Delta^{(k)} \neq 0, D(k) > \frac{k+1}{2} & \text{or} \\ \Delta^{(k)} \neq 0 \text{ and } D(k) = \frac{k+1}{2}, \text{ and B(k)=0} \end{cases}$$

$$\text{Case 2} = \begin{cases} \Delta^{(k)} \neq 0 \text{ and } D(k) < \frac{k+1}{2} & \text{or} \\ \Delta^{(k)} \neq 0,\ D(k) = \frac{k+1}{2}, \text{ and B(k)=1} \end{cases}$$

In case 1, we will define $\tau^{(k+1)}, \gamma^{(k+1)}$ by the equation (D3) and set

$$(D(k+1), B(k+1)) = (D(k), B(k))$$

In case 2, we will define $\tau^{(k+1)}, \gamma^{(k+1)}$ by the equation (D4) and set

$$(D(k+1), B(k+1)) = (k+1-D(k), 1-B(k))$$

It is easy to see that $B(k)$ only takes values 0 or 1. We have the following Propositions,

Proposition D.0.1 *: We always have (1): deg($\sigma^{(k)}$)$\leq D(k)$ with equality if B(k)=1. (2): deg($\omega^{(k)}$)$\leq D(k) - B(k)$ with equality if B(k)=0. (3): deg($\tau^{(k)}$)$\leq k - D(k)$ with equality if B(k)=0. (4): deg($\gamma^{(k)}$)$\leq k - D(k) - (1 - B(k))$ with equality if B(k)=1.*

Proof. See [2]. ∎

Proposition D.0.2 *: For each k, we have*

$$\bar{\omega}^{(k)}\tau^{(k)} - \sigma^{(k)}\gamma^{(k)} = x^k$$

Proof. See [2]. ∎

Proposition D.0.3 *: If σ and $\bar{\omega}$ are any pair of polynomials which satisfy*

$$\sigma(0) = 1 \quad \textit{and} \quad (1+S)\sigma = \bar{\omega} \textit{ mod } x^{k+1}$$

Let $D = max\{deg\ \sigma, deg\ \bar{\omega}\}$. Then there exist polynomials U and V such that

$$U(0) = 1, V(0) = 0$$
$$deg\ U \leq D - D(k), deg\ V \leq D - [k - D(k)]$$
$$\sigma = U\sigma^{(k)} + V\tau^{(k)}$$
$$\bar{\omega} = U\bar{\omega}^{(k)} + V\gamma^{(k)}$$

Proof. See [2]. ∎

Proposition D.0.4 *: If σ and $i\bar{\omega}$ are relative prime, and $\sigma(0) = 1$ and $(1 + S)\sigma = \bar{\omega}$ mod x^{k+1}, then*

(1) : *Either deg $\sigma \geq D(k) + 1 - B(k) \geq D(k)$, or deg $\bar{\omega} \geq D(k)$, or both*
(2) : *If deg(σ)$\leq \frac{k+1}{2}$ and deg($i\bar{\omega}$) $\leq \frac{k}{2}$, then $\sigma = \sigma^{(k)}$ and $\bar{\omega} = \bar{\omega}^{(k)}$.*

Proof. See [2]. ∎

We let $k = 2t$, then we inductively construct $\sigma, \bar{\omega}$ and finish the Berlekamp Algorithm.

Bibliography

Books

General

[1] *Schrödinger, E. W*hat is life. The Physical Aspect of Living Cell. Cambridge: At The University Press. New York: The MacMillan Company, 1945.

Coding theory

[2] *Berlekamp, E.R. A*lgebraic Coding Theory. New York: McGraw-Hill, 1968.

[3] *Berlekamp, E.R.(ed) K*ey Papers in The Development of Coding Theory. New York: IEEE press, 1974.

[4] *McEliece, R.J. The theory of Information and Coding.* Encyclopedia of Math. and its Aplications, Vol3, Reading, Mass: Addison-Wesley, 1977.

[5] *MacWilliams,J. and Sloane, N. The theory of Error-Correcting and Coding.* Amsterdam: North Holland Publishing Co. 1977.

[6] *Pretzel, O. C*odes and Algebraic Curves. Clarendon Press, Oxford.1998.

[7] *van Lint, J.H. I*ntroduction to Coding Theory. Springer. 1999

Commutative Algebra and Algebraic Geometry

[8] *Chevalley, C. Introduction to the theory of algebraic functions of one variable*, AMS, Providence, RI. 1951.

[9] *Hartshorne, R. Algebraic Geometry.* Springer-Verlag. 1977.

[10] *Kurosh, A.G. The theory of Groups*, Chelsea, New York, NY. 1955.

[11] *Mumford, D. Introduction to Algebraic Geometry.* Springer-Verlag. 1964.

[12] *Van der Waerden, B. Modern Algebra.* Frederick Unger Publ. Co. 1950.

[13] *Walker, R.J. Algebraic Curves*, Princeton University, Princeton, Dover reprint (1962)

[14] *Zariski-Samuel Commutative Algebra*, Vol I& II D. Van Nostrand Co. 1960.

Papers

Coding Theory

[15] *Berrou, C., and Glavioux, A., New Optimun Error Correcting Coding and Decoding: Turbo Codes.* IEEE Trans. Comm. vol 44, no.10,, pp 1261-1271. Oct. 1966.

[16] *Burton, H.O., and Weldon, E.J., Cyclic Product Codes.* IEEE Trans. Information Theory, It11 pp 433-439 9165.

[17] *Elias, P. Coding for Noisy Channels.* IRE Conv. Record, part 4. pp 37-46. Einhoven University of Technology. 1993

[18] *Duursma, I.M. Decoding codes from curves and cyclic codes.* Ph.D. dissetation, Einhoven University of Technology. 1993.

[19] *Feng,G.L. and Rao,T.R.N. A simple approach for construction of algebraic-geometric codes from affine plane curves.* IEEE Trans. on Information Theory 40, pp. 1003-1012. 1994.

[20] *Forney, G.D.Jr.. Generalized Minimum Distance Decoding.* IEEE Trans. on Information Theory vol 12, no. 2, pp. 125-131. Apr. 1966.

[21] *Gallager,R.G. Low-Density Parity-Check Codes.*. IRE Trans. on Info. Theory. vol. IT-8, pp21-28. Jqn. 1962.

[22] *Goppa,V.D. A new class of linear error-correcting codes.*. Problems of Information Transmission, 6. pp207-212. 1970.

[23] *Hartmann,C.R.P., and Tzeng, K.K. Generalizations of BCH bound.* Info. and Contral, 20, pp. 489-498. 1972.

[24] *Ihara, Y., Congruence relations and Shimura curves* Proc. symp. Pure Math. **33** part 2, 291-311. 1979.

[25] *Justensen, J., A Class of Constructive Asymptotically Good Algebraic Codes.* IEEE Trans. on Information Theory, 18 pp652-656, 1972.

[26] *Muller, D.E. Metric Properties of Boolean Algebra and Their Application to Switching Circuits.* Report No.46, Digital Computer Laboratory, Univ. of Illinois, April 1954.

[27] *Peterson, W.W., Encoding and Error-Correction Procedures for the Bose-Chaudhuri Codes.* IRE Trans. on Information Theory, Oct. 1960.

[28] *Reed, I.S. A Class of Multiple-Error-Correcting Codes and the Decoding Scheme.* J. IRE Transactions on Information Theory, September 1954.

[29] *Reed, I.S. and Solomon, G. Polynomial Codes Over Certain Finite Fields.* J. Soc. for Industrial and Applied Mathematics, June 1960.

[30] *Shannon, C.E. A mathematical theory of communication.* Bell Syst. Tech. J., 27. pp. 379-423, 623-656. 1948.

[31] *Skorobogatov, A.N. and Vlăduţ, S.C. On the decoding of algebraic-geometric codes.* IEEE Trans. on Information Theory 36, pp. 1051-1060. 1990.

[32] *Sugiyama, Y. Kasahara, M. Hirasawa, S. and Namekawa, T. A Method for Solving Key Equation for Decoding Goppa Codes.* Information and Control **27** 87-99, 1975.

[33] *Tsfasman, M.A., Vlăduţ, S.C., and Zink, T. Modular curves, Shimura curves and Goppa codes better then the Varshamov-Gilbert bound.*. Math. Nachr **109** 21-28. 1982.

[34] *Xing, C., Nonlinear Codes From Algebraic Curves Improving the Tsfasman-Vlă duṭ-Zink Bound.* IEEE Trans. on Information Theory, **Vol 49, No 7** 1652-1657. 2003.

Algebraic Geometry

[35] *Hasse, H. Zur Theorie der abstrakten elliptischen funktionenk: orper* I,II & III Crelle's Journel 1986 (175)

[36] *Serre, J.P. Sur le nombre des points rationnels d'une courbe algebraic sur un corps fini* C.R. Acad. Sc. Paris, 296, pp. 397-402. 1983.

[37] *Weil, A. Number of solutions of equations over finite field.* Bull. Amer. Math. Soc. 55 497-508i. 1949.

INDEX

Index