Ratio Mathematica Volume 39, 2020, pp. 79-110

On some computational and applications of finite fields

Jean Pierre Muhirwa[*]

Abstract

Finite field is a wide topic in mathematics. Consequently, none can talk about the whole contents of finite fields. That is why this research focuses on small content of finite fields such as polynomials computational, ring of integers modulo p where p is prime or a power of prime. Most of the times, books which talk about finite fields are rarely to be found, therefore one can know how arithmetic computational on small finite fields works and be able to extend to the higher order. This means how integer and polynomial arithmetic operations are done for $\mathbb{Z}_p$ such as addition, subtraction, division and multiplication in $\mathbb{Z}_p$ followed by reduction of p (modulo p). Only addition and multiplication arithmetic operations are considered for a small range of finite fields $(\mathbb{Z}_2 - \mathbb{Z}_{17})$. With polynomials, one can learn how arithmetic computational through polynomials over finite fields are performed as their coefficients are drawn from finite fields. The paper includes also construction of polynomials over finite fields as an extension of finite fields with polynomials i.e $F_q[x]/f(x)$, where $f(x)$ is irreducible over F_q. From the past decades, many researchers complained about the applications of some topics in pure mathematics and therefore the finite fields play more important role in coding theory, such as error-coding detection and error-correction as well as cyclic codes. Hence, this paper shows these applications.

Keywords: Finite Fields; Error-detection; Error-correction; Coding; Decoding; Codewords; Cosets; Syndromes.[1]

[*] University of Rwanda, College of Science and Technology, School of Science, Department of Mathematics, Kigali, Rwanda; muhijeapi@gmail.com.

[1] Received on January 20th, 2020. Accepted on June 19th, 2020. Published on December 31st, 2020. doi: 10.23755/rm.v39i0.521. ISSN: 1592-7415. eISSN: 2282-8214. ©Jean Pierre Muhirwa. This paper is published under the CC-BY licence agreement.

1 Introduction

The structure of this research paper includes the introductory part where some preliminary properties of set theory, group theory, ring theory and fields theory are discussed. In reality we can not know what is a field without defining a group and a ring since the field is a special case of the ring. Apart from introductory, the second section consist of computational in the first seven finite fields. The third, the fourth and the fifth parts of this paper discuss and compare the usual polynomial arithmetic computational and the finite field polynomial computational. The sixth part of this paper explains some of the applications of finite fields with the typical examples in coding and decoding theories, the seventh section gives the conclusion of the research paper while the last part acknowledges the financial support received from the Eastern Africa Universities Mathematics Programme-International Science Programme, University of Rwanda Node (EAUMP-ISP, UR-Node).

1.1 Preliminaries

Definition 1.1. *A set is a collection of distinct objects, considered as an object in it own rights. Sets are the one of the most fundamental concepts of mathematics.*

Example 1.1. *The set $\mathbb{R}$, denote the set of all real numbers, and this set includes rational numbers and irrational numbers (example π, $\sqrt{2}$, and e) $\mathbb{Z}$, denote the set of all integers for both sign (negative and positive).*

Definition 1.2. ***Group Theory**, a set $\mathbb{R}$ together with a binary operation is called a group if it satisfies the conditions such that closure, associative, admits identity element and inverse element under the operation within the elements of $\mathbb{R}$.*

Definition 1.3. ***Abelian Group**, a set $\mathbb{R}$ is an abelian group if it is a group for which commutative law within an operation together with $\mathbb{R}$ to the elements of $\mathbb{R}$ is verified.*

Definition 1.4. ***Ring Theory**, a set $\mathbb{R}$ together with two binary operations (addition and multiplication) on the elements of $\mathbb{R}$ is called a ring if the following conditions are satisfied:*

1. *$(\mathbb{R}, +)$ is an abelian group.*
2. *Associative law for multiplication and distributive law are also satisfied.*

Definition 1.5. ***Commutative Ring**, a commutative ring is a ring for which the multiplication is commutative.*

Definition 1.6. ***Commutative Ring with Unity***, *a commutative ring with unity is a ring for which there exists a non-zero multiplicative identity element.*

Example 1.2. *The set of integers $\mathbb{Z}$ is commutative ring with 1 as a multiplicative identity element.*

Definition 1.7. ***Field***, *a field is a commutative ring with unity and for which every non-zero element of that commutative ring is invertible.*

Example 1.3. *In the set of rational numbers, $\mathbb{Q}$, every non-zero element has its inverse i.e (Every non-zero element is invertible).*

Definition 1.8. ***Finite Field***, *a finite field is a field with a finite number of elements.*

Example 1.4. *Consider the set of integers modulo p ($\mathbb{Z}_p$), where p is prime integers). This set consists of $p - 1$ elements and all non-zero elements of this set are invertible.*

Definition 1.9. ***Galois Group***, *the Galois group of an extension of fields F/K, is the set of all automorphisms obtained by fixing the elements of K.*

Definition 1.10. ***Codewords***, *codewords are string of digits that can be interpreted by any machine as words or characters.*

Example 1.5. *The string* 100110 *is a codeword of the vector space $V(6, 2)$ of the length* 6 *over the finite field F_2.*

Definition 1.11. ***Prime Number***, *a prime number is a natural number that can be divisible only by 1 and itself (i.e, a prime number has two divisors namely 1 and the number itself).*

Example 1.6. *The first ten prime numbers are* $2, 3, 5, 7, 11, 13, 17, 19, 23, 29$.

Definition 1.12. ***Algorithm***, *an algorithm is a scientific term for solving an instance or a set of instructions that can be followed for solving a problem.*

Example 1.7. *To find the greatest common divisor (GCD) of two numbers a and b, we can apply division algorithm, and the GCD is the last non-zero remainder. All steps that are followed to determine the GCD will make an algorithm.*

1.2 Mathematical Definition of a Group

A set R together with a binary operation $(*)$ is said to be a group if it satisfies the following properties:

For a, b and $c \in R$,

1. $a * b \in R$ (closure)
2. $(a * b) * c = a * (b * c)$ (associativity)
3. There exists additive identity element e of R such that $a * e = e * a = a$, for all $a \in R$ (for (*) operation, identity is always e (identity element))
4. There exists inverse element a^{-1} of R such that $a * a^{-1} = a^{-1} * a = e$ (inverse element)
5. Furthermore if $a * b = b * a$, then R is said to be a commutative group or an abelian Group.

Note: this operation is not always $(*)$ it can be also addition, and it may be another operation defined on a set R.
However, in this research paper we are restricted on the usual addition and multiplication operators.

1.3 Mathematical Definition of a Ring

A set R together with two binary operations namely addition $(+)$ and multiplication $(*)$ is said to be a ring if the following 3 conditions are satisfied:
Fora, b and $c \in R$,

1. (R, +) must be an abelian group
2. $a * (b * c) = (a * b) * c$: associativity law for multiplication
3. $a * (b + c) = a * b + a * c$ (left distributive law) $(a + b) * c = a * c + b * c$ (right distributive law)

Note: The above two operations $(+)$ and $(*)$ are not necessarily the ordinary addition and multiplication operations, reason why the definition of these operations may be needed in mathematical expressions. But this paper considers them as ordinary addition and multiplication.

If there exists multiplicative identity element of R for each every non-zero element of R, always denoted 1 such that $a * 1 = 1 * a = a$, then we can call the ring R to be the ring with unity.

The inverse of an element a for the abelian group $(R, +)$ is denoted $(-a)$.

In addition if $a * b = b * a$, then R is called a commutative ring with unity. If every non- zero element of a commutative ring R with unity is invertible, then R

becomes a field.

1.4 Classification of fields

Fields can be classified by size or by the number of elements that a field possesses. If a field contains a finite number of elements then that field is called finite field, otherwise it is an infinite field. For the rest of the work we will proceed with the finite field only.

For example consider the commutative ring, $\mathbb{Z}_p$, where p is a prime number, is a commutative ring with unity which is the field hence finite field because it possesses finite number of element. This is the most popular example of finite field.

Then, definition of this topic as the name indicated above, a finite field is a field with a finite order (i.e number of elements is finite). It is also called Galois field (so named in honor of Evariste Galois). The order of a finite field is always a prime number or a power of a prime number. A finite field of order p^n is denoted $GF(p^n)$, often written as $F(p^n)$ in current usage.

$GF(p^n)$ is called the prime field of order p, where the p elements are denoted $0, 1, 2, 3, ..., p-1$. In the finite field $GF(p)$ if two elements are written as $a = b$ this is the same as $a \equiv b(mod \quad p)$. Finite fields are therefore denoted by $GF(p^n)$ instead of $GF(k)$ where $k = p^n$, for clarity. The finite field $GF(2)$ consists of elements $0, 1$ which satisfy the addition and multiplication modulo 2. Let us first consider the addition and multiplication of elements in $GF(2)$ as shown in following two tables below:

+	0	1
0	0	1
1	1	0

Table 1: The table shows the addition in $GF(2)$

*	0	1
0	0	0
1	0	1

Table 2: The table describes the multiplication in $GF(2)$

Clearly $GF(2)$ is finite field since it contains two elements 0 and 1 which is a finite number of elements and also by the rule that every non-zero element is invertible, in the table it is clear that 1 is the only non-zero element and it is invertible. The finite fields are classified by size, as follows:

1. The order or number of elements of finite fields is of the form p^n, where p is a prime number called the characteristic of the field, and n is a positive integer.
2. For every prime number p and a positive integer n, there exists a finite field with p^n elements.
3. Any two finite fields with the same number of elements are isomorphic. For example $\mathbb{Z}/(3)$ is isomorphic to F_3. That is under some renaming of the elements of one of these two fields, its addition and multiplication tables become identical to the corresponding tables of the other one. This classification is justified by using a naming scheme for finite fields that specifies only the order of the field.

Note: Finite fields are important and very useful in number theory, algebraic geometry, Galois Theory, cryptography, coding theory and quantum error correction. Its applications may also be appearing in the electrical circuits.

2 Computational Over Finite Fields with First seven Rings ($\mathbb{Z}_p$, where $p = 2, 3, 5, 7, 11, 13, 17$)

Arithmetic in a finite field is different from standard integers arithmetic. There are a limited number of elements in the finite field; all operations performed in the finite field result in an element within that field.

While each finite field is itself not infinite, there are infinitely many different finite fields; their number of elements (which is also called cardinality) is necessarily of the form p^n, where p is a prime number and n is a positive integer, and two finite fields of the same size are isomorphic. Consider $\mathbb{Z}/(3)$ is isomorphic to $\mathbb{Z}_3$. The prime p is called the characteristic of the finite field, and the positive integer n is called the dimension of the field over its prime field.

The finite field with p^n elements is denoted $GF(p^n)$ and is also called the Galois Field, in honor of the founder of finite field theory, Evariste Galois [Cox, 2011]. $GF(p)$, where p is a prime number, is simply the ring of integers modulo p. That is, one can perform operations (addition, subtraction, division and multiplication) by using the usual operation on integers, followed by reduction modulo p. For instance, in $GF(5)$, $4+3 = 7$ is reduced to 2 modulo 5. Division is multiplication by the inverse modulo p, which may be computed using the extended Euclidean algorithm.

A particular case is $GF(2)$, as addition and multiplication have been shown above in Table 1 and Table 2 respectively, and the only invertible element is 1. Now arithmetic operations in this paper are done on the first seven rings of integers modulo p ($\mathbb{Z}_p$), where p is a prime number, and those are $\mathbb{Z}_2, \mathbb{Z}_3, \mathbb{Z}_5, \mathbb{Z}_7, \mathbb{Z}_{11}, \mathbb{Z}_{13}$ and Z_{17}.

2.1 Arthmetic Operation in the Ring of Integers ($\mathbb{Z}_3$)

The class of residues in $\mathbb{Z}_3$ are $0, 1, 2$

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

Table 3: This is a table that shows the addition in $\mathbb{Z}_3$

*	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

Table 4: This table describes the multiplication in $\mathbb{Z}_3$

2.2 Arthmetic Operation in the Ring of Integers ($\mathbb{Z}_5$)

The class residues in $\mathbb{Z}_5$ are $0, 1, 2, 3, 4$

*	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Table 6: This a multiplication table in $\mathbb{Z}_5$

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Table 5: This is a table that illustrates how an addition is done in $\mathbb{Z}_5$

2.3 Arthmetic Operation in the Ring of Integers ($\mathbb{Z}_7$)

The class residues of $\mathbb{Z}_7$ are $0, 1, 2, 3, 4, 5, 6$

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

Table 7: An addition table in $\mathbb{Z}_7$

*	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

Table 8: Multiplication table in $\mathbb{Z}_7$

2.4 Arthmetic Operation in the Ring of Integers ($\mathbb{Z}_{11}$)

The class residues of $\mathbb{Z}_{11}$ are $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$

+	0	1	2	3	4	5	6	7	8	9	10
0	0	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10	0
2	2	3	4	5	6	7	8	9	10	0	1
3	3	4	5	6	7	8	9	10	0	1	2
4	4	5	6	7	8	9	10	0	1	2	3
5	5	6	7	8	9	10	0	1	2	3	4
6	6	7	8	9	10	0	1	2	3	4	5
7	7	8	9	10	0	1	2	3	4	5	6
8	8	9	10	0	1	2	3	4	5	6	7
9	9	10	0	1	2	3	4	5	6	7	8
10	10	0	1	2	3	4	5	6	7	8	9

Table 9: This table demonstrates the addition in $\mathbb{Z}_{11}$

*	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10
2	0	2	4	6	8	10	1	3	5	7	9
3	0	3	6	9	1	4	7	10	2	5	8
4	0	4	8	1	5	9	2	6	10	3	7
5	0	5	10	4	9	3	8	2	7	1	6
6	0	6	1	7	2	8	3	9	4	10	5
7	0	7	3	10	6	2	9	5	1	8	4
8	0	8	5	2	10	7	4	1	9	6	3
9	0	9	7	5	3	1	10	8	6	4	2
10	0	10	9	8	7	6	5	4	3	2	1

Table 10: Multiplication table in $\mathbb{Z}_{11}$

2.5 Arthmetic Operation in the Ring of Integers ($\mathbb{Z}_{13}$)

The class residues of $\mathbb{Z}_{13}$ are $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12$

+	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0	1	2	3	4	5	6	7	8	9	10	11	12
1	1	2	3	4	5	6	7	8	9	10	11	12	0
2	2	3	4	5	6	7	8	9	10	11	12	0	1
3	3	4	5	6	7	8	9	10	11	12	0	1	2
4	4	5	6	7	8	9	10	11	12	0	1	2	3
5	5	6	7	8	9	10	11	12	0	1	2	3	4
6	6	7	8	9	10	11	12	0	1	2	3	4	5
7	7	8	9	10	11	12	0	1	2	3	4	5	6
8	8	9	10	11	12	0	1	2	3	4	5	6	7
9	9	10	11	12	0	1	2	3	4	5	6	7	8
10	10	11	12	0	1	2	3	4	5	6	7	8	9
11	11	12	0	1	2	3	4	5	6	7	8	9	10
12	12	0	1	2	3	4	5	6	7	8	9	10	11

Table 11: This is an addition table in $\mathbb{Z}_{13}$

*	0	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10	11	12
2	0	2	4	6	8	10	12	1	3	5	7	9	11
3	0	3	6	9	12	2	5	8	11	1	4	7	10
4	0	4	8	12	3	7	11	2	6	10	1	5	9
5	0	5	10	2	7	12	4	9	1	6	11	3	8
6	0	6	12	5	11	4	10	3	9	2	8	1	7
7	0	7	1	8	2	12	3	10	4	11	5	12	6
8	0	8	3	11	6	1	8	4	12	7	2	10	5
9	0	9	5	1	10	6	2	11	7	3	12	8	4
10	0	10	7	4	1	11	8	5	2	12	9	6	3
11	0	11	9	7	5	3	1	12	10	8	6	4	2
12	0	12	11	10	9	8	7	6	5	4	3	2	1

Table 12: Multiplication Table in $\mathbb{Z}_{13}$

From this table, each non-zero element has its multiplicative inverse, the multiplicative inverse of 8 for example is 5, the multiplicative inverse of 11 is 6, and so on.

2.6 Arithmetic Operation in the Ring of Integers ($\mathbb{Z}_{17}$)

The class residues of $\mathbb{Z}_{17}$ are $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16$

+	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	0
2	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	0	1
3	3	4	5	6	7	8	9	10	11	12	13	14	15	16	0	1	2
4	4	5	6	7	8	9	10	11	12	13	14	15	16	0	1	2	3
5	5	6	7	8	9	10	11	12	13	14	15	16	0	1	2	3	4
6	6	7	8	9	10	11	12	13	14	15	16	0	1	2	3	4	5
7	7	8	9	10	11	12	13	14	15	16	0	1	2	3	4	5	6
8	8	9	10	11	12	13	14	15	16	0	1	2	3	4	5	6	7
9	9	10	11	12	13	14	15	16	0	1	2	3	4	5	6	7	8
10	10	11	12	13	14	15	16	0	1	2	3	4	5	6	7	8	9
11	11	12	13	14	15	16	0	1	2	3	4	5	6	7	8	9	10
12	12	13	14	15	16	0	1	2	3	4	5	6	7	8	9	10	11
13	13	14	15	16	0	1	2	3	4	5	6	7	8	9	10	11	12
14	14	15	16	0	1	2	3	4	5	6	7	8	9	10	11	12	13
15	15	16	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
16	16	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Table 13: This table points out how to perform an addition in $\mathbb{Z}_{17}$

*	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	0	2	4	6	8	10	12	14	16	1	3	5	7	9	11	13	15
3	0	3	6	9	12	15	1	4	7	10	13	16	2	5	8	11	14
4	0	4	8	12	16	3	7	11	15	2	6	10	14	1	5	9	13
5	0	5	10	15	3	8	13	1	6	11	16	4	9	14	2	7	12
6	0	6	12	1	7	13	2	8	14	3	9	15	4	10	16	5	11
7	0	7	14	4	11	1	8	15	5	12	2	9	16	6	13	3	10
8	0	8	16	7	15	6	14	5	13	4	12	3	11	2	10	1	9
9	0	9	1	10	2	11	3	12	4	13	5	14	6	15	7	16	8
10	0	10	3	13	6	16	9	2	12	5	15	8	1	11	4	14	7
11	0	11	5	16	10	4	15	9	3	14	8	2	13	7	1	12	6
12	0	12	7	2	14	9	4	16	11	6	1	13	8	3	15	10	5
13	0	13	9	5	1	14	10	6	2	15	11	7	3	16	12	8	4
14	0	14	11	8	5	2	16	13	10	7	4	1	15	12	9	6	3
15	0	15	13	11	9	7	5	3	1	16	14	12	10	8	6	4	2
16	0	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

Table 14: This a multiplication table in $\mathbb{Z}_{17}$

Apart from the 14 tables represented above, one may proceed in the same way up to the finite fields of $p - 1$ class residues with p being a prime number or a power of a prime number.

3 Arithmetic Computational of Polynomials over Finite Fields

The theory of polynomials over finite fields is important for investigating the algebraic structure of finite fields as well as for many applications. Above all, irreducible polynomials, the prime elements of polynomial rings over finite fields are indispensable for constructing finite fields and computing with the elements of finite fields [Rónyai, 1992].
A polynomial is an expression of the form $a_nx^n + a_{n-1}x^{n-1} + ... + a_1x + a_0$, for some non-negative integer n and where the coefficients $a_0, a_1..., a_n$ are drawn from some designated set S, which is in particular finite field and called the coefficient set.

Polynomial arithmetic deals with the addition, subtraction, multiplication, and division of polynomials.

3.1 What Problems Does Polynomial Arithmetic Adress?

Given two polynomials whose coefficients are derived from a set S, what can we say about the coefficients of the polynomial that results from an arithmetic operation on the two polynomials? If we insist that the polynomial coefficient all come from a particular S, then which arithmetic operations are permitted and which prohibited? Let us say that the coefficient set is a finite field F with its own rules for addition, subtraction, multiplication and division, and let us further say that when we carry out an arithmetic operation on two polynomials, we subject the operations on the coefficients to those that apply to the finite field F. Now what can be said about the set of such polynomials? All these questions will have their answers as we move on in this paper.

3.2 Ordinary Addition and Subtraction of Polynomials

Let $f(x) = a_2x^2 + a_1x + a_0$ and $g(x) = b_1x + b_0$ Then $f(x) + g(x) = a_2x^2 + (b_1 + a_1)x + (a_0 + b_0)$
Let $f(x) = a_2x^2 + a_1x + a_0$ and $g(x) = b_3x^3 + b_0$, Then $f(x) - g(x) = -b_3x^3 + a_2x^2 + a_1x + (a_0 - b_0)$

3.3 Ordinary Multiplication of Polynomials

Let $f(x) = a_2x^2 + a_1x + a_0$ and $g(x) = b_1x + b_0$, Then $f(x) * g(x) = a_2b_1x^3 + (a_2b_0 + a_1b_1)x^2 + (a_1b_0 + a_0b_1)x + a_0b_0$.

3.4 Ordinary Division of Polynomials

3.4.1 When is Division of Polynomials Permited?

Polynomial division is obviously not allowed for polynomials that are not defined over certain fields. For example, for polynomials defined over the set of all integers, you cannot divide $4x^2 + 5$ by the polynomial $5x$. If you tried, the first term of the quotient would be $(\frac{4}{5})x$ where the coefficient of x is not an integer. You can always divide polynomials defined over a certain field. What that means is that the operation of division is legal when the coefficients are drawn from a finite field. Note that, in general, when you divide such polynomial by another, you will end up with a remainder, and when, in general you divide one integer by another integer it is possible in purely integer arithmetic.

Therefore, in general, for polynomials defined over a field, the division of a polynomial $f(x)$ of a degree m by another polynomial $g(x)$ of a degree $n - m$ can be expressed by $f(x)/g(x) = q(x)g(x) + r(x)$, where $q(x)$ is the quotient and, $r(x)$ the remainder, so we can write for any two polynomials defined over a field, $f(x) = q(x) * g(x) + r(x)$ assuming that the degree of $f(x)$ is not less than that of $g(x)$. When $r(x)$ is zero, we say that $g(x)$ divides $f(x)$. This fact can also be expressed by saying that $g(x)$ is a divisor of $f(x)$ and by notation, $g(x)|f(x)$.

3.4.2 Division of a Polynomial by Another Upon Using Long Division

Let us divide the polynomial $8x^2 + 3x + 2$ by the polynomial $2x + 1$:

In this example, our dividend is $8x^2 + 3x + 2$ and the divisor is $2x + 1$. We now need to find the quotient.
Long division for polynomials consists of the following steps:

Step 1: Arrange both the dividend and the divisor in the descending powers of the variable.
Step 2: Divide the first term of the dividend by the first term of the divisor and write the result as the first term of the quotient.
In our example, the first term of the dividend is $8x^2$ and the first term of the divisor is $2x$, so the first term of the quotient is $4x$.
Step 3: Multiply the divisor with the quotient term just obtained and arranges the result under the dividend so that the same powers of x match up. Subtract the expression just laid out from the dividend. In our example, $4x$ times $2x + 1$ is equal to $8x^2 + 4x$. Subtracting this from the dividend yields $-x + 2$. consider the result of the above subtraction as the new dividend and go back to the first step. (The new dividend in our case is $(-x + 2)$. In our example, dividing the polynomial $8x^2 + 3x + 2$ by the polynomial $2x + 1$, yield quotient of $4x - 0.5$ and a remainder of 2.5.

3.5 Arithmetic Operations on Polynomials whose Coefficients Belong to a Defined Finite Fields

The arithmetic operations on polynomials whose coefficients are drawn from finite fields is not the same as the usual operations of polynomials. To see this, Let us consider the set of all polynomials whose coefficients belong to the finite field $\mathbb{Z}_7$ (which is the same as GF(7)). Here is an example of adding two such polynomials: $f(x) = 5x^2 + 4x + 6$, $g(x) = 5x + 6$ we get $f(x) + g(x) = 5x^2 + 9x + 12 =$

$5x^2 + 2x + 5$

If we perform the difference of both polynomials, $f(x) = 5x^2 + 4x + 6$ and $g(x) = 5x + 6$ then $f(x) - g(x) = 5x^2 - x = 5x^2 + 6x$ since the additive inverse of 5 in $\mathbb{Z}_7$ is 2 and that of 6 is 1. So $4x - 5x$ is the same as $4x + 2x$ and $6 - 6$ is the same as $6 + 1$, with both additions modulo 7.
The multiplication of polynomials $f(x) = 5x^2 + 4x + 6$, and $g(x) = 5x + 6$ is given by $f(x) * g(x) = 4x^3 + x^2 + 5x + 1$
Lastly the divison of polynomials $f(x) = 5x^2 + 4x + 6$, $g(x) = 2x + 1$ is given by $f(x)/g(x) = 6x + 6$. If you multiply the divisor $2x + 1$ with the quotient $6x + 6$, you get the dividend $5x^2 + 4x + 6$.

Let consider also the polynomials defined over $GF(2)$. Recall that the notation $GF(2)$ means the same thing as $\mathbb{Z}_2$. We are obviously talking about arithmetic modulo 2. First of all, $GF(2)$ is a sweet basic finite field. Recall that the number 2 is the first prime. (A prime has exactly two distinct divisors, 1 and itself). $GF(2)$ consists of the set $0, 1$. The two elements of this set obey the following addition and multiplication rules:

0 + 0 = 0
0 x 0 = 0
0 + 1 = 1
0 x 1 = 0
1 + 0 = 1
1 x 0 = 0
1 + 1 = 0
1 x 1 = 1

So the addition over $GF(2)$ is equivalent to the logical XOR operation, and multiplication to the logical AND operation. Some examples of polynomials defined over $GF(2)$: are $x^3 + x^2 - 1; -x^5 + x^4 - x^2 + 1; x + 1$, etc.

3.5.1 Arithmetic Computational of Polynomials Defined Over $GF(2)$

Here is an example of adding two such polynomials: $f(x) = x^2 + x + 1, g(x) = x + 1$, therefore $f(x) + g(x) = x^2 + 2x + 2 = x^2$

- Here is an example of subtracting two such polynomials, $f(x) = x^2 + x + 1, g(x) = x + 1$, then $f(x) - g(x) = x^2$
- Here is an example of multiplying two such polynomials, $f(x) = x^2 + x + 1$, and $g(x) = x + 1$, then $f(x) \times g(x) = x^3 + 1$

- Here is an example of dividing two such polynomials,$f(x) = x^2 + x + 1, g(x) = x + 1$, then $f(x)/g(x) = x$.

  If you multiply the divisor, $x + 1$ with the quotient x, you get $x^2 + x$. That when added to the remainder 1 gives us back the dividend $x^2 + x + 1$

3.6 Division of Polynomials Defined Over Finite Fileds

First, note that a polynomial is defined over a field if all its coefficients are drawn from that field. Dividing polynomials defined over a finite field is a little bit more frustrating than performing other arithmetic operations on such polynomials. Now your mental gymnastics must include both additive inverses and multiplicative inverses.Consider again the polynomials defined over $GF(7)$. Let's say we want to divide $5x^2 + 4x + 6$ by $2x + 1$. In a long division, we must start by dividing $5x^2$ by $2x$. This requires that we divide 5 by 2 in $GF(7)$. Dividing 5 by 2 is the same as multiplying 5 by the multiplicative inverse of 2. Multiplicative inverse of 2 is 4 since $2 \equiv 4$ mod 7 is 1. So we have $5 \equiv 2^{-1} = 5 \equiv 4 = 20$ mod $7 = 6$. Therefore, the first term of the quotient is 6x. Since the product of $6x$ and $2x + 1$ is $5x^2 + 6x$, we need to subtract $5x^2 + 6x$ from the dividend $5x^2 + 4x + 6$. The result is $(4 - 6)x + 6$, which (since the additive inverse of 6 is 1) is the same as $(4 + 1)x + 6$, and that is the same as $5x + 6$.

Our new dividend for the next round of long division is therefore $5x + 6$. To find the next quotient term, we need to divide $5x$ by the first term of the divisor, that is by $2x$. Reasoning as before, we see that the next quotient term is again 6. The final result is that when the coefficients are drawn from the set $GF(7))$, $5x^2 + 4x + 6$ divided by $2x + 1$ yields a quotient of $6x + 6$ with the remainder zero.
So we can say that as a polynomial defined over the field, $GF(7)$, $5x^2 + 4x + 6$ is a product of two factors, $2x + 1$ and $6x + 6$. We can therefore write $5x^2 + 4x + 6 = (2x + 1) \equiv (6x + 6)$

4 Irreducible Polynomials or Prime Polynomials

Definition 4.1. *According to [Rónyai, 1992], a polynomial $f \in F[x]$ is said to be irreducible over F (or irreducible in $F[x]$, or prime in $F[x]$) if f has positive degree and $f = g * h$, with $g, h \in F[x]$ implies that either g or f is a constant polynomial, otherwise it is reducible over F. The reducibility or irreducibility of a given polynomial depends heavily on the field under considerations. For instance, the polynomial $x^2 - 2 \in Q(x)$ is irreducible over the field Q of rational numbers, but $x^2 - 2 = (x + \sqrt{2})(x - \sqrt{2})$ but reducible over the field of real numbers*

$(\mathbb{R})$. *For polynomials over finite fields, the same argument hold except that the coefficients are reduced in mod p.*

Example 4.1. *$f(x) = x^2 + x + 1$ is irreducible over F_2 but $g(x) = x^2 + 1$ is reducible over F_2 to see this $g(x) = x^2 + 1 = (x + 1)(x + 1) = x^2 + 2x + 1$, since $2 \equiv 0 \quad mod(2)$, and then $2x \equiv 0 \quad mod(2)$. In few words we can say, when $g(x)$ divides $f(x)$ without leaving a remainder, we say $g(x)$ is a factor of $f(x)$. A polynomial $f(x)$ over a field F is called irreducible, if $f(x)$ cannot be expressed as a product of two polynomials, both over F and both of degree lower than that of $f(x)$. An irreducible polynomial is also referred to as a prime polynomial.*

5 Some Computational Tables of Quotient Polynomials Over Finite Fields

To represent the elements of an extension fields over finite fields in a computational table, we must have the quotient? $F_q[x]/f(x)$, where $f(x)$ is irreducible over $F_q[x]$. This form of polynomials are looked like powers of prime [Lidl and Niederreiter, 1994].

Example 5.1. *Let $f(x) = x^2 + 1 \in F_3[x]$. Thus to find the computational tables of $F_3[x]/(f(x))$, we need to find the residue class ring as p^n where n is the degree of polynomial $f(x)$, and then we have a set of residue class ring of $3^2 = 9$ elements, as it looks like a representation of $F(9)$, such as $0, 1, 2, x, 1 + x, 2 + x, 1 + 2x, 2x, 2 + 2x$, these are precisely the polynomials of degree less than 2 over F_3 by equating $x^2 + 1 = 0$ and this implies that $x^2 = -1 = 2$, but remember that computational in finite fields are followed by mod p [Gong et al., 2013]*

.

+	0	1	2	x	1+x	2+x	1+2x	2x	2+2x
0	0	1	2	x	1+x	2+x	1+2x	2x	2+2x
1	1	2	0	1+x	2+x	x	2+2x	1+2x	2x
2	2	0	1	2+x	x	1+x	2x	2+2x	1+2x
x	x	1+x	2+x	2x	1+2x	2+2x	1	0	2
1+x	1+x	2+x	x	1+2x	2+2x	2x	2	1	0
2+x	2+x	x	1+x	2+2x	2x	1+2x	0	2	1
1+2x	1+2x	2+2x	2x	1	2	0	2+x	1+x	x
2x	2x	2x+1	2+2x	0	1	2	1+x	x	2+x
2+2x	2+2x	2x	1+2x	2	0	1	x	2+x	1+x

Table 15: Addition table for $F_3[x]/(f(x))$

*	0	1	2	x	1+x	2+x	1+2x	2x	2+2x
0	0	0	0	0	0	0	0	0	0
1	0	1	2	x	1+x	2+x	1+2x	2x	2+2x
2	0	2	1	2x	2+2x	1+2x	2+x	x	1+x
x	0	x	2x	2	x+2	2x+2	x+1	1	2x+1
1+x	0	1+x	2+2x	x+2	2x	1	2	2x+1	x
2+x	0	2+x	1+2x	2x+2	1	x	2x	x+1	2
1+2x	0	1+2x	2+x	x+1	2	2x	x	2x+2	1
2x	0	2x	x	-2	2x+1	x+1	2x+2	x	x+2
2+2x	0	2+2x	1+x	2x+1	x	2	1	x+2	2x

Table 16: Multiplication Table for $F_3[x]/(f(x))$

6 Applications of Finite Fields

6.1 Algebraic Coding Theory

It is one of the major applications of finite field. This theory has its origin in famous theorem of Shannon that guarantees the existence of codes that can transmit information at rates close to the capacity of a communication channel with an arbitrary small probability of error. One of the purposes of algebraic coding theory, the theory of error-correcting and error-detecting codes is to devise methods for construction of such codes [von zur Gathen et al.]. During the last two decades more and more abstract algebraic tools such as the theory of finite fields and the theory of polynomials over finite fields have influenced coding. In particular, the description of redundant codes by polynomials over F_q is a milestone in this development. The fact that one can use shift registers for coding and decoding establishes a connection with linear recurring sequences. In our discussion of algebraic coding theory we do not consider any of the problems of the implementation or technical realization of the codes. We restrict ourselves to the study of basic properties of block codes and the description of some interesting classes of block codes.

6.1.1 Linear coding

The problem of communicating the information, in particular the coding and decoding of information for the reliable transmission over a ”noisy” channel is of great importance today. Typically, one has to transmit a message which consists of finite string of symbols that are elements of some finite alphabet. For instance, if this alphabet consists of simply 0 and 1, the message can be described as binary number.

Generally the alphabet is assumed to be finite fields. Now the transmission of finite string of elements of the alphabet over a communication channel need not to be perfect in the sense that each bit of information is transmitted unaltered over this channel. As there is no ideal channel without ”noise” the receiver of the transmitted message may obtain distorted information and may make errors in interpreting the transmitted signal.

One of the main problems of coding theory is to make the errors, which occur for instance because of noisy channel, extremely improbable.

The methods of improve the reliability of transmission depend on properties of finite fields. A basic idea in algebraic coding theory is to transmit redundant information together with the message one wants to communicate; that is, one extends the string of message symbols to a longer string in a systematic way.

A simple model of communication system is shown in the figure bellow:

We assume that the symbols of the message and the coded message are elements of the same finite field F_q. Coding means to encode a block of k message symbols $a_1, a_2, ..., a_k$ where $a_i \in F_q$ into a code word $c_1, c_2, ..., c_n$ of n symbols, where $c_j \in F_q$, with $n > k$. We regard the code word as an n-dimensional row vector $c \in F_q^n$. Thus f in the Figure below is a function from F_q^k into F_q^n, called a coding scheme, and $g : F_q^n \to F_q^k$ is a decoding scheme.

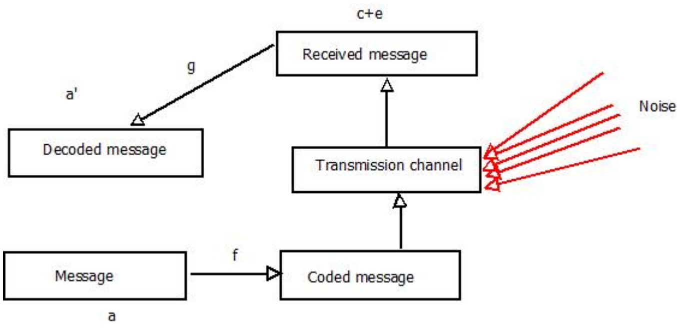

Figure 1: Communication figure that shows how a message is coded, transmitted and decoded

A simple type of coding scheme arises when each block $a_1a_2...a_k$ of message symbols is encoded into a code word of the form $a_1a_2...a_kc_{k+1}...c_n$, where the first k

symbols are the original message symbols and the additional $n-k$ symbols in F_q are control symbols. Such coding schemes are often presented in the following way. Let H be a given $(n-k)\times n$ matrix with entries in F_q that is of the special form $H=(A, I_{n-k})$, where A is $(n-k)\times k$ matrix and I_{n-k} is the identity matrix of order $n-k$. The control symbols $c_{k+1},...,c_n$ can then be calculated from the system of the equations $HC^T=0$, for code word c. The equations of this system are called parity-check equations. The examples of this theory will be given later.

6.2 Error-Correcting Codes (Practice of Linear Code)

Since the theory of codes was developed in order to ensure reliability of transmitted information, as an example, consider the ISBN (International Standard Book Number) of published book. This number usually appears on the back of the book in the bottom right-hand corner. The ISBN consists of a nine-digits $0, 1, ..., 9$ or the symbol X (standing for 10). This final symbol may be calculated from the other nine as follows:
From an integer N by adding together the first digit, twice the second digit, three times the third and so on. The check digit is the remainder when N is divided by 11. For example, a book with first 9 digits 019853453 will have $N = 0+2+27+32+25+18+28+40+27 = 199$, and so the check digit should be 1, giving ISBN 01953453 1. The point about such a number is that if it is inaccurately copied, and an error is made in any of the digits in the first nine locations (such as the last ”5” being copied as a ”3”), then the resulting number will not have ”1” as its check digit. This is an example of error-detecting code: the ISBN detects when a single error is made after transcribing the number. Another example of finding check digit is that of 102463798, then $N=1\times1+0\times2+2\times3+4\times4+6\times5+3\times6+7\times7+9\times8+8\times9=264$, and divide this number by 11 to get the check digit which is 0, and hence giving ISBN 102463798 0 In this part we shall explain methods which not only detect errors, but also enables us to correct it.

Definition 6.1. *Let p be prime integer. Denote by $V(n,p)$ the set of all sequences of length n of the elements from the set $\mathbb{Z}_p$ of congruence classes modulo p, so that $V(n,p)$ has p^n elements. We will usually omit the commas and brackets commonly used to denote elements of the vector spaces, so that* $(1,0,1)$, *will be written as* 101. *Thus $V(3,2)$ consists of the eight sequences* $000, 001, 010, 011, 100, 101, 110, 111$ *while $V(2,3)$ consists of the nine sequences* $00, 01, 02, 10, 11, 12, 20, 21, 22$. *We add sequences by adding the corresponding terms, by just remembering that we are adding congruence classes. Thus, for example in* $V(3,2)$, $110+011=101$

while in $V(2,3)$, $12+11=20$. We can also multiply an element in $V(n,p)$ by a congruence class by multiplying each term in the sequence by the representative for the congruence class and reducing modulo p. For example, in the space $V(3,3)$ we see that 2(102) =201. In fact $V(n,p)$ is a vector space of dimension n over the field $\mathbb{Z}_p$

Definition 6.2. *A linear (n,k)-code is any k-dimensional subspace C of the vector space $V(n,p)$. Thus C satisfies the following two conditions:*
The difference of any two elements of C is an element of C, and the product of any element of C with an element of $\mathbb{Z}_p$ is also an element of C. The elements of C are called codewords.

Note: A subspace of a vector space is necessary non-empty, so condition (1) ensures that the zero element of the vector space is in the subspace C. It then follows by the additive version that C is a group under addition.

Example 6.1. *Consider the four elements $000, 001, 010, 011$ of $V(3,2)$. These are precisely the four sequences which start with 0. This subspace of $V(3,2)$ satisfies condition one, that subtracting any two of these gives a sequence starting with 0. Also condition (2) holds, since 0 and 1 are the only elements of $\mathbb{Z}_2$ and then multiply each sequence by any of these two elements we get an element starting with 0. Therefore the four elements form a linear $(3,2)$-code.*

Definition 6.3. *Let v be any element of $V(n,p)$. The weight of v is the number of non-zero terms in the sequence v. If v and w are two elements of $V(n,p)$, the distance $d(v,w)$ is the number of places at which v and w differ.*

Example 6.2. *In $V(4,3)$ the weight of 1201 is three, since there are three non-zero entries. The distance from 1201 to 2211 is two, since these two vectors differ in two places. In $V(5,5)$ the weight of 13402 is four and so on.*

Proposition 6.1. *Let u, v and w be any elements of $V(n,p)$. Then*

1. $d(u,v) \geq 0$ with equality if and only if $u = v$;
2. $d(u,v) = d(v,u)$; and
3. $d(u,v) + d(v,w) \geq d(u,w)$.

Proof.

1. It follows directly from the definition that $d(u, v)$ is positive except u and v do not differ anywhere.

2. This is always true for u and v.

3. In each location at which u and w differ, v cannot agree with both u and w. Thus every contribution to the value of $d(u, w)$ provides a contribution to either $d(u, v)$ or to $d(v, w)$. □

Definition 6.4. *Let C be subspace of $V(n, p)$. The minimum distance d of C is the least distance between different codewords: $d = \min_{u,v}\{d(u, v)\}$. The next result shows that for a linear code, the minimum distance d can be calculated from the code words.*

Proposition 6.2. *Let C be a linear $(n - k)$-code. Then the minimum distance of C is equal to the smallest possible weight of any non-zero codeword.*

Proof.

Let f be the smallest possible weight of any non-zero codeword, and let 0 denote the sequence consisting entirely of zeros. Suppose that w is a codeword of weight f. Then $d(w, 0)$ if and only if so $f \geq d$. Now let u and v be pair of codewords with $d(u, v) = d$. Since C is a linear code, the word $u - v$ is a codeword of weight d, so $d \geq f$. It follows that $d = f$.

The importance of the minimum distance lies in the detecting the errors and correction of those errors. To see this, consider the following proposition. □

Proposition 6.3. *Let C be linear code with minimum distance d. Then C detects $d - 1$ or fewer errors, and corrects e errors for any e with $2e + 1 \leq d$.*

Proof.

Let v be a vector which has distance f from a codeword c, where $f \leq d - 1$. We think of c as the transmitted word and v as the received word, so that there are f errors in transmission. Since d is the minimum distance for C, the received v cannot be a codeword. We express this by saying that the code C detects d or fewer errors. Suppose now that v has distance e from a codeword c and also that $2e + 1 \leq d$. Then there can be no other codeword near to v: If c_1 was in C and $d(v, c_1) \leq e$, then by property of triangle inequality $d(c, c_1) \leq d(c, v) + d(v, c_1) \leq e + e < d$, which contradicts the definition. Thus there is a unique nearest codeword to v, and we say that C corrects e errors in this case. □

Definition 6.5. *Let n and k be any positive integers with $n > k$. Let p be a prime number. A (standard) generator matrix G over $\mathbb{Z}_p$ is a $k \times n$ matrix with entries in $\mathbb{Z}_p$, in which the first k columns form an identity $k \times k$ matrix. Given such a matrix, we obtain a linear code by regarding the rows as sequences and taking all possible linear combinations of these. Alternatively, we can consider the code as consisting of all sequences obtained from matrix multiplications of the form $u.G$ as u varies over all sequences of length k over $\mathbb{Z}_p$.*

Example 6.3. *Consider the generator matrix over $\mathbb{Z}_2$*

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

The corresponding code consists of the combinations of the rows and so has four elements: $000; 101; 011$ *and* 110. *The codewords can also be described as the vectors of the form uG, as u varies over the four vectors* $00; 01; 10; 11$. *Every non-zero codeword has weight* 2, *so the codeword detects one error, but does not correct errors. For example, 111 is not among codewords (so it is detected) but it is of equal distance from the two codewords 101 and 011 in G, so it cannot be corrected.*

Example 6.4. *Another example of a binary code (code over $\mathbb{Z}_2$) is provided by the matrix*

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

There are 8 code words obtained from the rows of this matrix:
$000000; 100110; 010101; 110011; 001011; 101101; 011110; 111000.$

There are four code words of weight 3, three code words of weight 4 and one of weight 0. The minimum distance (d) of this code is therefore 3, so the code detects $d - 1$ errors means two errors and corrects one error. For example, 100111 lies at distance one from a unique codeword, 100110 and so there is unique way to correct one error. The vector 100001, however has distance two from 000000 and 110011, so cannot be corrected.

Example 6.5. *Consider the following generator matrix over $\mathbb{Z}_3$:*

$$G = \begin{pmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 2 \end{pmatrix}$$

In this case, the codeword consists of the linear combinations of the rows of the matrix, including multiplication by 1 and 2 since $p = 3$. There are 9 code words: $0000; 1021; 2012; 0112; 1100; 2121; 0221; 1212$ *and* 2200.

Since there is a codeword of weight 2, this code detects one error. Note that the minimum distance is 2 despite the fact that each row of the generator matrix has weight 3.

Example 6.6. *Consider also the following important code over $\mathbb{Z}_3$*

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 2 & 2 \\ 0 & 0 & 1 & 0 & 0 & 0 & 2 & 1 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 & 0 & 0 & 2 & 2 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 2 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

By considering this matrix, the minimum distance of this code is at most 5 since there is a row of the generator matrix of weight 5. It can be shown that the minimum distance is exactly 5, so that code corrects two errors. This is the Ternary Golay code and is one of the most important code. More details and its descriptions are found in [Cohen et al., 2013].

We now consider the problem of decoding a linear (n, k)-code C. This is done by listing the left cosets of the subgroup C of $V(n, p)$ in a table known as the cosets decoding table. The table is organized by writing the codewords as its first row with the zero codeword first. Each subsequent row is a left coset of C. The entries in the first column are the coset representatives, now called cosets leaders. The algorithm for choosing the r^{th}-coset leader is to choose any word of minimum weight not already included in the first $(r - 1)$ rows. Then to decode a given vector, locate it in the table, and correct it to the codeword standing in the same column of the coset decoding table.

Example 6.7. *Consider Example 6.6, above there are eight code words which form a subgroup C of the vector space $V(6, 2)$. Since V has $2^6 = 64$ elements,* *this subgroup has index 64/8=8. To form a complete coset decoding table, we list the elements of C in a row. We then choose any element v_2 which is of smallest weight among those not in the first row and write this at the left hand end of the second row. The second row is obtained by adding each element of C in turn to this. Thus the second row is just the coset of C with respect to v_2. Continue this process by choosing v_3 to be of the smallest weight among the elements not in the first two rows, and so on. This process is not unique, but depends upon the choice of coset representatives [Pless, 1998]. One example of these choices is given in the following table*

000000,	*100110,*	*010101,*	*110011,*	*001011,*	*101101,*	*011110,*	*111000*
100000,	*000110,*	*110101,*	*010011,*	*101011,*	*001101,*	*111110,*	*011000*
010000,	*110110,*	*000101,*	*100011,*	*011011,*	*111101,*	*001110,*	*101000*
001000,	*101110,*	*011101,*	*111011,*	*000011,*	*100101,*	*010110,*	*110000*
000100,	*100010,*	*010001,*	*110111,*	*001111,*	*101001,*	*011010,*	*111100*
000010,	*100100,*	*010111,*	*110001,*	*001001,*	*101111,*	*011100,*	*111010*
000001,	*100111,*	*010100,*	*110010,*	*001010,*	*101100,*	*011111,*	*111001*
100001,	*000111,*	*110100,*	*010010,*	*101010,*	*001100,*	*111111,*	*011001*

To decode any element v of $V(6,2)$, we locate v in the table and then correct it to the element in the first row of the column containing v. Thus to use the table to decode 011010*, we need to locate it (it is in the fifth row and seventh column) and correct it to the element in the first row and the same column, giving* 011110*. Note that the cosets representative for the last row is not easy to find. According to the algorithm, we need a word of weight* 2 *not in the first seven rows. The representative we choose,* 100001*, is not unique. This is actually a somewhat cumbersome way to arrange the decoding, since an exhaustive search is required. The calculation can be made more systematic for codes given by (standard) generator matrices using (standard) parity check matrices [Sayed, 2011].*

Definition 6.6. *Let C be an (n, k)-linear code over $\mathbb{Z}_p$ defined using $k \times n$ generator matrix G of the form*

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

where, A is $k \times (n-k)$ matrix. The parity check matrix associated with G is the $(n-k) \times n$ matrix

$$P = \begin{pmatrix} & 1 & 0 & 0 & ... & 0 \\ -A^T & 0 & 1 & 0 & ... & 0 \\ & \vdots & & & & \\ & 0 & 0 & 0 & ... & 1 \end{pmatrix}$$

Note: The generator matrix G above is often written, in a block matrix form as $G = (I_k|A)$. similarly, the parity check matrix is written as $P = (-A^T|I_{(n-k)})$ [kar, 2012].

Example 6.8. *The parity check matrix of the generator matrix over $\mathbb{Z}_2$. The parity check of the matrix of*

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

is the matrix

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

[kar, 2012]. This matrix is obtained by considering the matrix

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

and then after computing $-A^T$ get the above matrix P given by $-A^T|I_{(n-k)}$ [kar, 2012].

Definition 6.7. *Let C be a linear (n, k)-code with generator matrix G and associated parity matrix P. For any v in $V(n, p)$, let v^T denote the transpose of v, the column vector obtained by writing the members of the sequence v vertically. Then the syndrome of v is the element of $V(n - k, p)$ given by Pv^T. Thus in the above example, the syndrome of $v = 100000$ is 110 and the syndrome of $v = 110011$ is* 000.

Note: If C is a code with standard parity check matrix P, then an element v in $V(n, p)$ is a codeword if and only if the syndrome of v is the zero sequence.

Example 6.9. *We need not store or the complete coset decoding table, but merely a table of two columns, the coset representatives and their syndromes. In our previous example in which P was*

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

This table would be as the following,

Coset representatives	Syndromes
000000	000
100000	110
010000	101
001000	011
000100	100
000010	010
000001	001
100001	111

Table 17: This a table of syndromes and cosets representatives

Thus to decode a given vector such as 100111, calculate its syndrome to obtain 001. This is the syndrome for the seventh row, so this vector is not a codeword, but the word 100110 obtained by subtracting 000001 is a codeword. The advantage of listing coset representatives together with syndromes is that, it is much easier to find any missing coset representatives, since each sequence in $V(n-k,p)$ occurs as syndrome. Thus in this above example, the syndrome for the last row must be 111 *because the other seven sequences of length 3 have already been used as syndromes. This enables us to find a representative relatively easily (compared with searching through the first seven rows), by seeing how to combine known coset leaders and their syndromes to obtain 111.*

6.3 Cyclic Codes

Definition 6.8. *In the paper of [Peterson and Brown, 1961], a linear code C is called a cyclic code if it has the following property:*

If $(c_0, c_1, c_2, ..., c_{n-1}) \in C$, then it is also reality that $(c_1, c_2, ..., c_{n-1}, c_0) \in C$. From this definition the automorphism group Aut(C) of a code C is the set of permutations $\delta \in S_n$ such that $\delta(c) \in C$ for all $c \in C$, where $\delta(c_0, c_1, c_2, ..., c_{n-1}) =$

$(c_{\delta(0)}, ..., c_{\delta(n-1)})$. *In other words, the code, C is cyclic if and only if the permutation $\delta = (0, 1, 2, ..., n-1)$ is in Aut (C)[Roberts and Vivaldi, 2005].*

Example 6.10. *Let C be a subspace of a vector space $V(6, 7)$ and consider the code words $v = (345601)$ of C, then C is cyclic code if $(4, 5, 6, 0, 1, 3)$; $(5, 6, 0, 1, 3, 4)$; $(6, 0, 1, 3, 4, 5)$; $(0, 1, 3, 4, 5, 6)$; $(1, 3, 4, 5, 6, 0)$, all are elements of C. We can define an algebraic structure by looking at cyclic code if we let C to be a cyclic code over the field F_q and we set $R_n \doteqdot F_q[x]/(x^n - 1)$. We can take the elements of R_n as polynomials of degree at most $n-1$ over F_q, where multiplication can be happen except that $x^n = 1, x^{n+1} = x$, and so on. From this, we can deduce one to one correspondence between polynomials and the code words of cyclic code as can be seen in [Sziklai, 2013].*

Example 6.11. *Let C be a subspace of a vector space $V(5, 7)$ over $F_7 = \mathbb{Z}/7\mathbb{Z}$ and let consider the code word $(1, 2, 3, 5, 6)$. Then we can find the polynomial of degree less than 5 correspond to this code word which is given by $1 + 2x + 3x^2 + 5x^3 + 6x^4$. To find the elements of $R_n \doteqdot F_q[x]/(x^n - 1)$, we do it as found for the previous case of quotient finite fields, and these are precisely the polynomials of degree at most $n - 1$, hence the total number of the elements of $R_n \doteqdot F_q[x]/(x^n - 1)$, are q^n elements.*

Example 6.12. *Let find the elements of $R_3 \doteqdot F_2[x]/(x^3 - 1)$, here our $q = 2$ and $n = 3$, therefore the total number of the elements of this polynomial field are $q^n = 2^3 = 8$ polynomials of degree less than 3 whose coefficients are in F_2. So the elements $R_3 \doteqdot F_2[x]/(x^3 - 1)$ are $0, 1, x, 1 + x, x^2, x^2 + 1, 1 + x + x^2, x + x^2$.*

Theorem 6.1. *From this kind of cyclic codes we define also an ideal of R_n given by $I_C \doteqdot (c(x) \doteqdot c_0 + c_1x + ... + c_{n-1}x^{n-1}) \in R_n c \doteqdot (c_0, c_1, ..., c_{n-1}) \in C$*

Proof.

Let $c, d \in I_C$, $a \in R_n$, then we want to show that $c - d \in I_C$ and $ac \in I_C$, therefore $c(x) = c_0 + c_1x + ... + c_{n-1}x^{(n-1)}, d(x) = d_0 + d_1x + ... + d_{(n-1)}x^{(n-1)}$ and $a(x) = a_0 + a_1x + ... + a_{n-1}x^{(n-1)}$. So, $c(x) - d(x) = c_0 - d_0 + (c_1 - d_1)x + ... + (c_{n-1} - d_{n-1})x^{(n-1)} \in I_C \Rightarrow (c_0 - d_0, c_1 - d_1, ..., c_{n-1} - d_{n-1}) \in C$.
$C \in I_C \Leftrightarrow (c_0, c_1, ..., c_{n-1}) \in C$
$d \in I_C \Leftrightarrow (d_0, d_1, ..., d_{n-1}) \in C$.
$(c_0 - d_0, c_1 - d_1, ..., c_{n-1} - d_{n-1}) \in C$, is a code word of cyclic code C (since C is a vector space of $V(n, q)$ over F_q. It remains to show that $a(x)c(x)$ is an element of I_C. Then $a(x)c(x) = (a_0 + a_1x + ... + a_{n-1}x^{(n-1)})(c_0 + c_1x + ... + c_{n-1}x^{(n-1)}) = a_0c_0 + a_0c_1 + a_0c_2 + ... a_1c_0 + a_1c_1 + ... + a_2c_0 + ...$, is also a code word of length $n-1$.

Let illustrate by using example, let $a(x) \in R_3 \doteqdot F_2[x]/(x^3 - 1)$, and $c(x) \in I_C$, we have $a(x) = a_0 + a_1x + a_2x^2$ and $c(x) = c_0 + c_1x + c_2x^2$, where $a_i \in F_q$ for $i = 0, 1, 2$ and $c_i \in C$ for $i = 1, 2, 3$. Then $a(x)c(x) = (a_0+a_1x+a_2x^2)(c_0+c_1x+c_2x^2) = a_0c_0+(a_0c_1+a_1c_0)x+(a_0c_2+a_1c_1+a_2c_0)x^2+(a_1c_2+a_2c_1)x^3+(a_2c_2)x^4$. But $x^3 = 1$ and $x^4 = x$, then we have $a_0c_0 + a_1c_2 + a_2c_1 + (a_0c_1 + a_1c_0 + a_2c_2)x + (a_0c_2 + a_1c_1 + a_2c_0)x^2 \in I_C$.

$\Rightarrow (a_0c_0 + a_1c_2 + a_2c_1; a_0c_1 + a_1c_0 + a_2c_2; a_0c_2 + a_1c_1 + a_2c_0) \in C$.
$\Rightarrow (a_0c_0, a_0c_1, a_0c_2) + (a_1c_2, a_1c_0, a_1c_1) + (a_2c_1, a_2c_2, a_2c_0)$.
$\Rightarrow a_0(c_0, c_1, c_2) + a_1(c_2, c_0, c_1) + a_2(c_1, c_2, c_0)$.
But $(c_0, c_1, c_2), (c_2, c_0, c_1), (c_1, c_2, c_0) \in C$ since C is cyclic code. Therefore I_C is an ideal of R_n. □

Theorem 6.2. *Let I_C be an ideal of $R_(n)$ and let $g(x) \in C$ be monic polynomial of minimal degree $l = deg(g(x))$. Then*

a. $g(x)$ is the only monic polynomial of degree l in I_C.

b. $g(x)$ generates I_C as an ideal of R_n.

Proof.

Let f be any other non- zero monic polynomial of minimal of I with degree less than l then $f - g \in I$, but $f \neq g \Rightarrow f - g \neq 0$, $f(x) - g(x) = c_kx^k + ... + c_1x + c_0$ and this polynomial is not monic, it becomes monic if we divide it by c_k^{-1} with $c_k \neq 0$, and then we get $1/c_k(f(x)-g(x)) = x^k+...+d_1x+d_0$, where $d = c_i/c_k$ for $i = 0, 1, ..., k$. Hence $k < l$ which contradicts that l is the minimal degree. Therefore, $g(x)$ is unique monic polynomial of the minimal degree. $g(x)$ generates I means that $I =< g >= gh, h \in R_n$, this also means if $f \in I$, then $f = gh$ for some $h \in R_n$. Let $f \in I \subset R_n = F_q[x]/(x^n - 1)$, write $f(x) = g(x)q(x) + r(x) \in I$ with $deg(r(x)) < deg(g(x)) = l$.
$\Leftrightarrow f(x) - g(x)q(x) = r(x) \in I$ (since $q(x), g(x) \in I$).
$\Rightarrow r(x) = 0$
$\Rightarrow f(x) = g(x)q(x)$
$\Rightarrow f \in< g >$ and $I \in< g >$ But $g \in I$, so $< g >\in I$. Hence $I =< g >$. □

7 Conclusion

This paper has discussed about finite fields whereby some important definitions, propositions, theorems and their proofs have been given in order to capture what finite fields are and how finite fields deal with operations in different ways from usual known operations that may be performed for a set of integers. The operations procedure required any arithmetic followed by reduction of p, and this is the reason why several tables from finite fields $\mathbb{Z}_2$ to $\mathbb{Z}_{17}$ are computed to highlight how one may compute in finite fields. It includes polynomials arithmetic operations over finite fields such as addition, subtraction, multiplication, and division. The arithmetic polynomials over finite fields are computed by using the reduction of p to its coefficients, because their coefficients are drawn from finite fields that are taken into consideration. Besides polynomials computational over finite fields, this paper also explains what are cyclic codes and their applications. This research paper has further shown the applications of finite fields in the most important domain of communication regarding algebraic coding theory, code error-detection and error-correction, whereby coding and decoding schemes using cosets representative and syndromes table are discussed by using tangible examples. From this paper one may learn about finite fields and its applications and be able to extend up to $p-1$ class residues with p being any prime number or any power of a prime number.

8 Acknowledgement

I would like to acknowledge the financial support from the Eastern Africa Universities Mathematics Programme-International Science Programme, University of Rwanda Node (EAUMP-ISP, UR-Node).

References

Parity check matrix recognition from noisy codewords. *arXiv preprint arXiv:1205.4641*, 2012.

Arjeh M Cohen, Hans Cuypers, and Hans Sterk. *Some tapas of computer algebra*, volume 4. Springer Science & Business Media, 2013.

David A Cox. *Galois Theory.*, volume 61. John Wiley & Sons, 2011.

Guang Gong, Katalin Gyarmati, Fernando Hernando, Sophie Huczynska, Dieter Jungnickel, Gohar M Kyureghyan, Gary McGuire, Harald Niederreiter, Alina Ostafe, and Igor E Shparlinski. *Finite fields and their applications: character sums and polynomials*, volume 11. Walter de Gruyter, 2013.

Rudolf Lidl and Harald Niederreiter. *Introduction to Finite Fields and their Applications*. Cambridge University Press, 2 edition, 1994. doi: 10.1017/CBO9781139172769.

William Wesley Peterson and Daniel T Brown. Cyclic codes for error detection. *Proceedings of the IRE*, 49(1):228–235, 1961.

Vera Pless. *Introduction to the theory of error-correcting codes*, volume 48. John Wiley & Sons, 1998.

John AG Roberts and Franco Vivaldi. Signature of time-reversal symmetry in polynomial automorphisms over finite fields. *Nonlinearity*, 18(5):2171, 2005.

Lajos Rónyai. Galois groups and factoring polynomials over finite fields. *SIAM Journal on Discrete Mathematics*, 5(3):345–365, 1992.

Mohamed Sayed. Coset decomposition method for decoding linear codes. *International Journal of Algebra*, 5(28):1395–1404, 2011.

Péter Sziklai. *Applications of polynomials over finite fields*. PhD thesis, ELTE TTK, 2013.

Joachim von zur Gathen, Igor E Shparlinski, and Henning Stichtenoth. Finite fields: Theory and applications.