SOME RESULTS ON APN FUNCTIONS AND WEAKLY REGULAR BENT FUNCTIONS OVER FINITE FIELDS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

İLKSEN ACUNALP ERLEBLEBİCİ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
CRYPTOGRAPHY

JULY 2023

Approval of the thesis:

SOME RESULTS ON APN FUNCTIONS AND WEAKLY REGULAR BENT FUNCTIONS OVER FINITE FIELDS

submitted by **İLKSEN ACUNALP ERLEBLEBİCİ** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Cryptography Department, Middle East Technical University** by,

Prof. Dr. A. Sevtap Selçuk-Kestel _____
Dean, Graduate School of **Applied Mathematics**

Assoc. Prof. Dr. Oğuz Yayla _____
Head of Department, **Cryptography**

Assoc. Prof. Dr. Oğuz Yayla _____
Supervisor, **Cryptography, METU**

Examining Committee Members:

Prof. Dr. Murat Cenk
Cryptography, METU _____

Assoc. Prof. Dr. Oğuz Yayla
Cryptography, METU _____

Assoc. Prof. Dr. Ahmet Sınak
Mathematics and Computer Science,
Necmettin Erbakan University _____

Prof. Dr. Zülfükar Saygı
Mathematics, TOBB ETU _____

Assoc. Prof. Dr. Fatih Sulak
Mathematics, Atılım University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: İLKSEN ACUNALP ERLEBLEBİCİ

Signature :

ABSTRACT

SOME RESULTS ON APN FUNCTIONS AND WEAKLY REGULAR BENT FUNCTIONS OVER FINITE FIELDS

Acunalp Erleblebici, İlksen
Ph.D., Department of Cryptography
Supervisor : Assoc. Prof. Dr. Oğuz Yayla

July 2023, 80 pages

In this thesis, we present our work on Carlet's bivariate APN construction and representations of APN power functions. We report necessary and sufficient conditions on some families of bivariate APN functions and our observations about the representation of the APN power functions by following the methods given by Calderini et al. in 2021 and Budaghyan et al. in 2022 and we give some computational results on this research. On the other hand, we study the application of cryptographic functions in coding theory, and we derive several linear codes with good parameters by employing weakly regular bent functions over the finite fields of odd characteristics.

Keywords: APN functions, bivariate APN construction, APN power functions, linear code, weight distribution, weakly regular bent function

ÖZ

SONLU CİSİMLER ÜZERİNDE APN VE ZAYIF DÜZENLİ BÜKÜK FONKSİYONLARA İLİŞKİN BAZI SONUÇLAR

Acunalp Erleblebici, İlksen
Doktora, Kriptografi Bölümü
Tez Yöneticisi : Doç. Dr. Oğuz Yayla

Temmuz 2023, 80 sayfa

Bu tezde Carlet'in iki değişkenli neredeyse mükemmel lineer olmayan (APN) fonksiyonlarının ve APN kuvvet fonksiyonlarının üzerine yaptığımız çalışmalar sunulmaktadır. 2021'de Calderini ve arkadaşlarının, 2022'de Budaghyan ve arkadaşlarının verdiği yöntemlerden yararlanılarak sırasıyla bazı iki değişkenli APN ailelerinin inşası için gerekli ve yeterli koşullar ve APN kuvvet fonksiyonlarının gösterimi hakkındaki incelemelerimiz sunulmaktadır ve bu araştırma hakkında bazı hesaba dayalı sonuçlar verilmektedir. Diğer taraftan, kriptografik fonksiyonların kodlama teorisindeki uygulamasını çalışıyoruz, ve tek karakteristikli sonlu cisimler üzerinde zayıf düzenli bükük fonksiyonları kullanarak iyi parametrelere sahip birçok doğrusal kod elde ediyoruz.

Anahtar Kelimeler: Neredeyse mükemmel lineer olmayan (APN) fonksiyonlar, iki değişkenli APN fonksiyonların inşası, APN kuvvet fonksiyonları, doğrusal kod, ağırlık dağılımı, zayıf düzenli bükük fonksiyon

to loving memory of my dad

ACKNOWLEDGMENTS

First and foremost, I would like to express my very great appreciation to my thesis supervisor Assoc. Prof. Dr. Oğuz Yayla for his patient guidance, enthusiastic encouragement and valuable advice during the development and preparation of this thesis. In addition to his professional guidance, I want to extend my appreciation for his personal support and friendship.

Also, I would like to express my endless gratitude to Assoc. Prof. Dr. Ahmet Sınak for his valuable suggestions, encouragement and time. His contribution has been truly remarkable for me.

I would like to express my sincere gratitude to all my friends, especially Banu and Betül, who supported me throughout this process and made life more beautiful.

I would like to express my profound gratitude to my parents, Sadet and Ahmet, and my sister, Sema, whose unwavering support and boundless love have been a constant source of strength and comfort in my life.

I would like to give my special thanks to my husband, Volkan, for his support, not only in this study but in every facet of our life together and for providing me with the extra strength to complete my thesis.

I would like to thank my daughter Eylül and my son Ediz for the meaning, unconditional love and joy they brought to my life.

TABLE OF CONTENTS

ABSTRACT vii

ÖZ ix

ACKNOWLEDGMENTS xi

TABLE OF CONTENTS xiii

LIST OF TABLES xvii

LIST OF FIGURES xviii

CHAPTERS

1 INTRODUCTION 1

1.1 Motivation 2

1.2 Contribution 4

1.3 Organization 4

2 PRELIMINARIES 7

2.1 Vectorial Boolean Functions 7

2.1.1 Vectorial Boolean Functions and Block Ciphers . . 7

2.1.2 Representations of Vectorial Boolean Functions . . 12

2.1.3 Differential Uniformity and Nonlinearity of Vectorial Boolean Functions 15

2.1.4 Equivalence Relations on Vectorial Boolean Functions 18

2.1.4.1 EA-equivalence 19

2.1.4.2 CCZ-equivalence 20

2.1.4.3 Cyclotomic equivalence 20

2.1.5 APN Power Functions 21

2.1.6 Known Infinite Families of Non-power APN Functions 23

2.2 Cyclotomic Fields 26

2.3 Linear Codes 27

2.4 Weakly regular bent functions 28

3 APN FUNCTIONS SPECIFIC TO CARLET'S BIVARIATE APN CONSTRUCTION 31

3.1 Three Cases of APN Functions From Carlet's Construction . 32

3.2 Biprojective Almost Perfect Nonlinear Functions 36

3.2.1 Family f_1 37

3.2.2 Family f_2 39

3.2.3 Family f_3 40

4 REPRESENTATIONS OF APN POWER FUNCTIONS 43

4.1 Dobbertin Power Function's Representations 43

4.2 Computational Results of Dobbertin Power Function's Representations 46

4.3 Computational Results of Niho Power Functions Representations 48

5 LINEAR CODES GENERATED FROM WEAKLY REGULAR BENT FUNCTIONS OVER $\mathbb{F}_p$ 53

5.1 The construction methods of linear codes from functions . . . 53

5.2 Three-weight linear codes based on the set $\mathcal{D}_1$ 55

5.3 Two-weight linear codes based on the set D_2 58

5.4 Two-weight linear codes based on the set $\mathcal{D}_0$ 63

6 CONCLUSION 67

REFERENCES 69

APPENDICES

A SAGEMATH IMPLEMENTATIONS OF ALGORITHMS IN THIS STUDY 75

CURRICULUM VITAE 79

LIST OF TABLES

Table 2.1 Known APN power functions x^d over $\mathbb{F}_2^n$ 22

Table 2.2 Infinite families of quadratic APN polynomials over the finite field $\mathbb{F}_{2^n}$ that are well-known 25

Table 5.1 The Hamming weights of $\mathcal{C}_{\mathcal{D}_1}$ in Theorem 9 when m is even 57

Table 5.2 The Hamming weights of $\mathcal{C}_{\mathcal{D}_1}$ in Theorem 9 when m is odd 58

Table 5.3 The Hamming weights of $\mathcal{C}_{\mathcal{D}_2}$ in Theorem 10 when m is odd with $p \equiv 3 \pmod 4$ or m is even 62

Table 5.4 The Hamming weights of $\mathcal{C}_{\mathcal{D}_2}$ in Theorem 10 when m is odd with $p \equiv 1 \pmod 4$ 62

Table 5.5 The Hamming weights of $\mathcal{C}_{\mathcal{D}_0}$ in Theorem 11 65

LIST OF FIGURES

Figure 2.1 Block Cipher [26] 8

Figure 2.2 SPN [26] 10

Figure 2.3 Feistel Cipher [26] 10

Figure 2.4 AES [26] 11

Figure 4.1 The location of the representations for $m = 2$, $k = 1, 2, 3$ 49

Figure A.1 Representation of Dobbertin Function by combining a function with weight 3 and inverse of a function with weight 2 76

Figure A.2 Representation of Dobbertin Function by combining a function with weight 4 and inverse of a function with weight 3 76

Figure A.3 Representation of Niho Function by combining a function with weight 2 and inverse of a function with weight 3 77

Figure A.4 Representation of Niho Function by combining a function with weight 3 and inverse of a function with weight 4 77

CHAPTER 1

INTRODUCTION

Vectorial boolean functions, which accept binary sequences of length n as input and produce binary sequences of length m as output (with n and m being positive integers), hold significant significance in the realm of cryptography within the field of computer science. The security and effectiveness of cryptographic systems are closely tied to the ability of these functions to withstand various types of cryptographic attacks. Moreover, they find wide-ranging utility in diverse domains, including sequence design, combinatorics, coding theory, and projective geometry. Indeed, differential uniformity and nonlinearity are two important cryptographic properties of functions. APN functions [1] are the vectorial boolean functions with perfect differential uniformity. In this thesis we consider Carlet's bivariate APN construction method [19]. After presenting this construction, Zhou and Pott in [65] construct a new APN family, and Taniguchi revealed a new APN family by simplifying Carlet's construction in [59]. This construction is extended By Calderini et al. in 2021 [18].

Linear codes find diverse applications in various fields, including graph theory, authentication codes, data storage systems, communication, consumer electronics, design theory and cryptography. Linear codes with few weights, in particular, hold significant practical value and find application in a wide range of systems. Numerous construction methods exist for generating such codes, and one of the prominent approaches is based on functions defined over finite fields.

Constructing linear codes from functions has been a well-explored research area in the literature. While substantial progress has been made in this direction, it remains an active area of study. Many linear codes have been derived from cryptographic

functions, including quadratic functions [27, 28, 31], weakly regular bent functions [27, 28, 39, 50, 58, 61], almost perfect nonlinear functions [22, 47, 64], and weakly regular plateaued functions [51, 52, 55]. These code constructions based on functions play a crucial role in various applications and continue to be of interest in both theory and practice.

The first part of this thesis focuses on bivariate APN functions, when utilized as S-boxes in block ciphers or as non-linear combining or filtering functions in the pseudo-random generators of stream ciphers, demonstrating robust performance against both differential and linear cryptanalysis. In the second part of the thesis, we consider the APN power family, especially Dobbertin and Niho APN power family and deal with the different representations of these functions. In the third part of this thesis, we deal with few-weight linear codes based on weakly regular bent functions over finite fields.

1.1 Motivation

Given positive integers n and m, vectorial Boolean functions or (n, m)-functions are mappings between the vector spaces $\mathbb{F}_2^n$ and $\mathbb{F}_2^m$, where $\mathbb{F}_2$ is the finite field with two elements and they are implemented in a cryptosystem as substitution boxes (S-boxes) in block ciphers in order to create confusion, and nonlinear combining or filtering functions in the pseudo-random generators of stream ciphers. S-boxes must be resistant to differential cryptanalysis [2]. Differential cryptanalysis involves studying how variations in the input can influence the resulting differences in the output. Let F be any function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$, F is called differentially δ-uniform if the equation $D_aF(x) = F(x) + F(x + a) = b$ has at most δ solutions for every nonzero $a \in \mathbb{F}_2^{n*}$ and every $b \in \mathbb{F}_2^m$. If x_1 is a solution of the equation then $x_1 + a$ is also a solution. So the smallest possible value of differential uniformity is 2, then there is a minimum limit of 2, as $\delta \geq 2$. The function's resistance to differential cryptanalysis depends on its differential uniformity being small. Almost perfect nonlinear functions (APN) have $\delta = 2$ and they are resistant to differential cryptanalysis.

APN functions have a significant place in symmetric cryptography. These functions

can be constructed using various methods, resulting in an infinite family of such functions. Since 1990, only a small number of infinite families of APN functions have been investigated. There are six known infinite families of APN monomials or APN power functions seen on Table 2.1, represented by $F(x) = x^d$ over F_{2^n} where d is positive integer. Due to their great resilience to differential attacks and simple hardware implementation, APN power functions with low differential uniformity have been the subject of much research in recent years. There are also about 15 known infinite families of quadratic APN polynomials presented in [18]. This specifically demonstrates how difficult it is to create such functions. Among the known APN functions, the APN functions with exceptional power hold a prominent position due to their significance. For this reason, we study their different representations. These investigations will shed light on the cryptographic properties of these functions.

In a finite field $\mathbb{F}_p$ with p elements, where p is an odd prime, an $[n, k, d]$ linear code C is a k-dimensional linear subspace of $\mathbb{F}_p^n$, defined over $\mathbb{F}_p$, with a Hamming distance of d. Linear codes are a fundamental component of modern cryptographic systems. They provide the necessary tools for error correction, data integrity, secure key exchange, and protection against various attacks, making them indispensable for securing sensitive information in a digital world.

The construction of linear codes from functions has been a thoroughly investigated research domain in existing literature. Two primary, generic construction methods, referred to as the *first* and *second* methods, can be distinguished from other approaches in the literature for constructing linear codes from functions. Many linear codes with exceptional parameters have been generated using cryptographic functions based on both the *first* generic construction method (as demonstrated in [22, 28, 50, 51]) and the *second* generic construction method (as shown in [28, 31, 52]). We note that these codes find themselves application areas in secret sharing and information theory. To be motivated by this, in this thesis, we use the second generic construction method on weakly regular bent functions to produce good linear codes.

1.2 Contribution

Carlet introduced in [19] bivariate APN construction, in which APN functions are built by using the simplest Maiorana–McFarland function. In the first part of this thesis, we work on Carlet's APN function construction method. First of all, by using the methods introduced by Calderini et al in [18], we give new necessary and sufficient conditions for some APN functions built by using Maiorana–McFarland and bi-projective functions in Carlet's construction. Then in the second part of this thesis inspired by [9], we report our observations about the representation of APN power functions and give our experimental data on it. When we come to the third part, we give new linear code with flexible parameters by employing weakly regular bent functions.

Based on the research presented in this thesis, we have the opportunity to share our findings with the academic community through an international conference presentation and the submission of two research papers. These papers are titled:

- İ. Acunalp Erleblebici, O. Yayla, *A General Version of Carlet's Construction of APN Functions*, 27th International Conference on Applications of Computer Algebra, 2022, Gebze, Türkiye.
- İ. Acunalp Erleblebici, A. Sınak, O. Yayla, *On the Carlet's Bivariate APN Construction and Representation of Dobbertin Power Functions*, eprint
- İ. Acunalp Erleblebici, A. Sınak, O. Yayla, *Few-weight linear codes based on weakly regular bent functions over finite fields*, eprint

1.3 Organization

In the next chapter, we go over some fundamental concepts, and then in Chapter 3, we introduce APN functions specific to Carlet's construction introduced in [19], and extended in [18]. In Chapter 4, we give our observations about representations of APN power functions, extend the results introduced in [9] and give our experimental results on these representations. In Chapter 5, we report new few-weight linear codes

based on weakly regular bent functions over finite fields.

CHAPTER 2

PRELIMINARIES

2.1 Vectorial Boolean Functions

Consider a positive integer, n. $\mathbb{F}_{2^n}$ denotes the finite field comprising 2^n elements, while $F_{2^n}^*$ signifies the set containing its non-zero elements, forming its multiplicative group. A Boolean function, denoted as f, is a function that accepts a sequence of 0s and 1s as input and produces either 0 or 1 as output. Furthermore, a vectorial Boolean function represents an extension of traditional Boolean functions. The functions from $\mathbb{F}_{2^n}$ to $\mathbb{F}_{2^m}$ are called vectorial Boolean function or (n, m)-functions. Given such a function F, the Boolean functions $f_1, \ldots, f_m$ such that

$$F(x) = F(x_1, \ldots, x_n) = (f_1(x), \ldots, f_m(x))$$

are referred to as F's coordinate functions. The component functions of F are the linear combinations of these coordinate functions with non-zero coefficients. When the values of m and n are left unspecified, functions labelled as (n, m)-functions are commonly known as vectorial Boolean functions or S-boxes. These S-boxes serve as the nonlinear components in block ciphers, as described in reference [26], in Figure 2.1

2.1.1 Vectorial Boolean Functions and Block Ciphers

In a block cipher, plaintext P is divided into blocks P_i of a specific length, and multiple blocks are encrypted using the same encryption key. Block ciphers operate iteratively, constructed by applying a reversible transformation referred to as a round

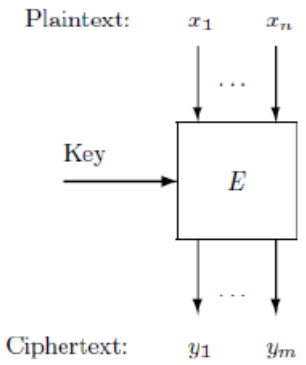

Figure 2.1: Block Cipher [26]

function repeatedly.

The composition of N round functions can be written as

$$E_K(.) = F^0_{K_0} \circ F^1_{K_1} \circ \cdots \circ F^{N-1}_{K_{N-1}}(.).$$

Here, $F^i_{K_i}$ represents the i-th round function, and K_i corresponds to the i-th round key. These round keys are derived from the original key K through the key schedule algorithm. Substitution Permutation Networks (SPN) in Figure 2.2 and Feistel Networks in Figure 2.3 are among the most well-known and widely used designs for iterated block ciphers.

An SPN's round function typically consists of three essential layers:

- Substitution Layer: This layer applies a substitution operation to the data, often using an S-box or a similar nonlinear transformation to introduce confusion.
- Permutation Layer: In this layer, the positions of the data bits are rearranged or permuted to enhance diffusion.
- Key Addition Layer: The round key is XORed (added) to the data to introduce the effect of the encryption key.

These three layers work in combination to achieve the cryptographic operations within a single round of an SPN-based block cipher.

The substitution layer, which is the only nonlinear component of the cipher, is composed of several S-boxes and vectorial Boolean functions are what S-boxes are. The

properties of vectorial Boolean functions, particularly the S-boxes used in Substitution-Permutation Networks (SPNs), play a crucial role in the security and functionality of SPN-based block ciphers.

- Bijectiveness of S-Boxes: For decryption to be possible, the S-boxes (vectorial functions) should be bijective. This means that each output value should have a unique corresponding input value, ensuring that the transformation can be reversed during decryption.
- Permutation Layer Linearity: The permutation layer in an SPN is typically designed as a linear transformation. This linear diffusion helps in spreading input differences across the block, enhancing security.
- Bit-Wise Key Addition: In the key addition layer, the round key K_i is added bit-wise to the internal state. This process introduces the effect of the encryption key on the data.
- Security Implications: The properties of the vectorial Boolean functions used in SPNs directly impact the cipher's security. Properties such as nonlinearity, resistance to cryptanalysis techniques like differential and linear attacks, and their algebraic properties are crucial in determining the cipher's strength against attacks.

The design and properties of the vectorial Boolean functions, as well as the overall structure of SPNs, are fundamental considerations in the development of secure block ciphers. These aspects impact both the security and the ability to perform decryption in the context of SPN-based ciphers.

The substitution-permutation cipher used by the Advanced Encryption Standard (AES) [44] (in Figure 2.4) has S-boxes that are the inverse functions.

Indeed, linear and differential attacks [44] are two of the most potent techniques used to analyze and potentially break block ciphers. The functional characteristics that measure the resistance of vectorial Boolean functions, including S-boxes used in block ciphers, to these attacks, are:

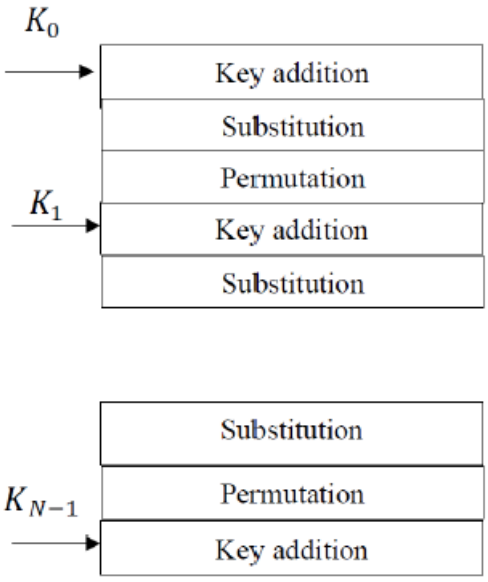

Figure 2.2: SPN [26]

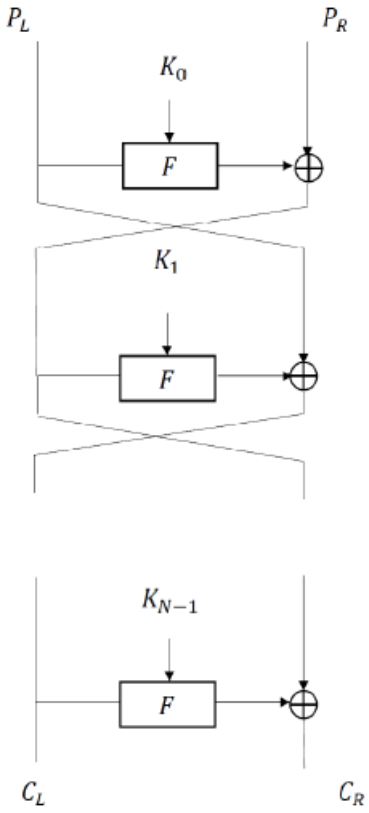

Figure 2.3: Feistel Cipher [26]

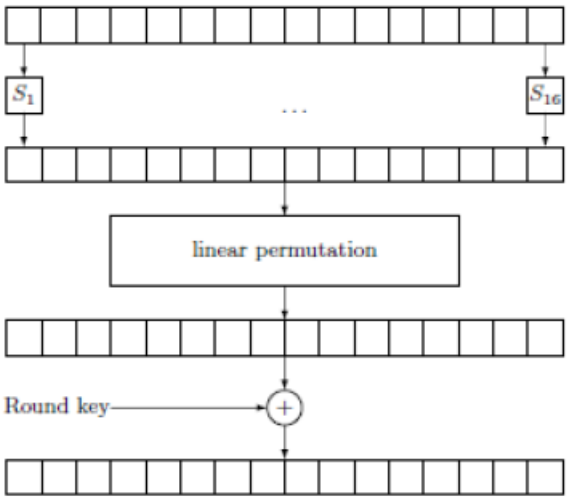

Figure 2.4: AES [26]

1. Differential Uniformity: Differential uniformity is a property that quantifies how differences in input bits propagate to differences in output bits through the S-box or vectorial Boolean function. Lower differential uniformity indicates greater resistance to differential attacks, as it makes it harder for attackers to predict the relationship between input and output differences.

2. Nonlinearity: Nonlinearity measures how closely the function resembles a linear transformation. Higher nonlinearity is desirable because it makes it more difficult for attackers to exploit linear approximations in linear attacks.

In addition to these characteristics, there are two other essential requirements for vectorial Boolean functions:

1. Large Algebraic Degree: Vectorial Boolean functions should have a sufficiently high algebraic degree. A high algebraic degree is crucial to protect against higher-order differential attacks, which can exploit polynomial relations between inputs and outputs to break the cipher.

2. Balancedness: Balancedness ensures that there is no statistical dependence between the inputs and outputs of the function. This property helps prevent attackers from identifying patterns or biases in the function's behavior, which could be exploited in statistical attacks.

These characteristics and requirements collectively contribute to the overall security of block ciphers, making them resistant to various types of attacks, including differential and linear attacks, as well as higher-order differential and statistical attacks.

2.1.2 Representations of Vectorial Boolean Functions

The algebraic normal form (ANF) of any (n, m)-function F is represented as:

$$\sum_{I \subseteq \{1,\ldots,n\}} a_I (\prod_{i \in I} x_i) \, ; a_I \in \mathbb{F}_2^m$$

In this representation, F is expressed as a sum of terms, where each term is associated with a binary vector x, and a_I represents coefficients in the finite field $\mathbb{F}_2^m$.

Relation for Coefficients a_I:

$$a_I = \sum_{x \in F_2^n | supp(x) \subseteq I} F(x)$$

This relation states that the coefficient a_I is obtained by summing the values of the function F over all binary vectors x in $\mathbb{F}_2^n$ where the support (positions with 1s) of x is a subset of I.

Relation for the Function $F(x)$:

$$F(x) = \sum_{I \subseteq supp(x)} a_I$$

This relation expresses the function $F(x)$ as the sum of coefficients a_I, where I varies over all subsets of the support of x.

These relations provide a way to understand and compute the coefficients a_I and the function $F(x)$ in terms of each other, based on the properties of the ANF representation. These relationships are useful in analyzing and working with Boolean functions and algebraic normal forms in various cryptographic and computational applications.

Indeed, the algebraic degree of a function, as defined, is a crucial measure of its complexity in terms of the algebraic normal form (ANF). Specifically, it represents the highest degree of monomials (terms) with nonzero coefficients in the ANF.

For cryptographic purposes, it is vital for vectorial functions to have high algebraic degrees:

- Functions with higher algebraic degrees are more resistant to various cryptographic attacks, including the Berlekamp-Massey attack on stream ciphers and higher-order differential attacks on block ciphers.
- High algebraic degree increases the difficulty for attackers to find algebraic relations between the function's inputs and outputs. This makes it harder for them to break the encryption scheme through algebraic analysis.
- Functions with high algebraic degrees tend to be more nonlinear, which is a desirable property in cryptography. Nonlinearity helps thwart linear and differential attacks, as it introduces complexity into the function's behavior.

In summary, a high algebraic degree is a key requirement for cryptographic vectorial functions because it contributes to the overall security of cryptographic systems by making them more resistant to a range of potential attacks.

When the dimensions of the input and output spaces of a vectorial Boolean function are equal (i.e., $m = n$), a second representation known as the univariate representation can be employed. In this representation, any vectorial Boolean function F can be uniquely expressed as a polynomial in the polynomial ring $\mathbb{F}_{2^n}[X]$ using Lagrange interpolation. The polynomial's degree is at most $2n - 1$, and it is represented as:

$$F(X) = \sum_{0 \leq i < 2^n} A_i X^i,\ A_i \in \mathbb{F}_{2^n}.$$

Here, each coefficient A_i is an element from the finite field $\mathbb{F}_{2^n}$, and the polynomial $F(X)$ captures the behavior of the vectorial Boolean function F over the binary vector X. This univariate representation is a powerful way to analyze and work with vectorial Boolean functions, especially when the input and output dimensions are the same ($m = n$). It allows for the application of polynomial algebra techniques to study and manipulate these functions.

When we assume $n = 2m$, it implies that the dimension of the input space is twice the dimension of the output space. In this context, it's possible to identify the finite field $\mathbb{F}_{2^n}$ with the direct product of $\mathbb{F}_{2^m}$ and itself, i.e., $\mathbb{F}_{2^m} \times \mathbb{F}_{2^m}$.

This identification means that elements in the field $\mathbb{F}_{2^n}$ can be represented as ordered pairs (a, b), where a and b are elements from $\mathbb{F}_{2^m}$. This is useful in various mathematical and cryptographic contexts, especially when dealing with vectorial Boolean functions and operations involving fields of characteristic 2.

By making this identification, you can often simplify mathematical expressions and computations when working with vectorial Boolean functions in situations where $n = 2m$. Then a function$F : F_{2^n} \to F_{2^n}$ can be represented as a (univariate) polynomial in $F_{2^n}[X]$, or a (bivariate) polynomial in $F_{2^m}[x, y] \times F_{2^m}[x, y]$.

A linear map $L : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ is an $\mathbb{F}_2$-linear vector space endomorphism of $\mathbb{F}_{2^n}$. The univariate representation of a linear map L takes the form:

$$L(X) = \sum_{0 \leq i < n} B_i X^{2^i},\ B_i \in \mathbb{F}_{2^n}.$$

This representation is known as a linearized polynomial, and its algebraic degree is at most one.

An affine map $A : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ is defined as $A(X) = L(X) + C$ where L is linear map and $C \in \mathbb{F}_{2^n}$ is a constant. Affine maps combine linear transformations with translations.

Quadratic functions are a subset of (n, m)-functions with an algebraic degree at most 2. They can be represented as:

$$Q(X) = \sum_{0 \leq i \neq j < n} A_{i,j} X^{2^i + 2^j} + L(X) + C, A_{i,j} \in \mathbb{F}_{2^n},$$

where L is linear map and $C \in \mathbb{F}_{2^n}$ is a constant. Quadratic functions encompass more complex transformations compared to linear or affine functions.

These definitions and representations are essential in the study and analysis of vectorial Boolean functions, particularly in the context of cryptographic algorithms where understanding the algebraic properties and degrees of these functions is crucial for security and performance analysis.

For m/n, define trace function $Tr_n^m = \mathbb{F}_{2^n} \to \mathbb{F}_{2^m}$ as $Tr_n^m(x) = \sum_{i=0}^{n/m-1} x^{2^{mi}}$. If $m = 1$, Tr_n^1 is denoted by Tr_n.

2.1.3 Differential Uniformity and Nonlinearity of Vectorial Boolean Functions

Differential uniformity and nonlinearity are two important properties used to assess the cryptographic strength of vectorial Boolean functions, particularly in the context of symmetric key cryptography and the design of cryptographic algorithms.

Differential Uniformity: Differential uniformity measures how differences in input bits propagate to differences in output bits when applying a vectorial Boolean function. It quantifies the uniformity of these differences. In cryptographic applications, lower differential uniformity is desired because it makes it harder for attackers to predict the relationship between input and output differences. Vectorial Boolean functions with low differential uniformity are more resistant to differential cryptanalysis, which is an attack method that analyzes how small input differences affect output differences.

Nonlinearity: Nonlinearity assesses how closely a vectorial Boolean function resembles a linear transformation. It quantifies the deviation from linearity. In cryptography, higher nonlinearity is preferred because it introduces complexity into the function's behaviour, making it harder for attackers to exploit linear approximations in attacks. Nonlinearity is crucial in resisting linear cryptanalysis, which is an attack technique based on linear approximations of the cipher's behaviour.

Both properties are essential for ensuring the security of cryptographic algorithms, as they help resist various types of cryptanalysis attacks. Cryptographers often strive to design vectorial Boolean functions with low differential uniformity and high nonlinearity to enhance the security of the cryptographic systems in which they are employed.

Absolutely, vectorial Boolean functions play a critical role in the design of cryptographic algorithms, particularly in the construction of substitution boxes, commonly referred to as S-boxes. These S-boxes are integral components in the development of block ciphers.

Vectorial Boolean functions, particularly when used in the form of S-boxes, are essential components in the creation of secure block ciphers. They add complexity,

nonlinearity, and resistance to attacks, all of which are vital for the strength and reliability of cryptographic algorithms used to protect sensitive data and communications. Due to the fact that $\mathbb{F}_{2^n}$ can be viewed as an n-dimensional vector space over the prime field $\mathbb{F}_2$, $\mathbb{F}_{2^n}$ can be associated with $\mathbb{F}_2^n$, this is allowed. Indeed, in many cryptographic contexts, finite field elements in $\mathbb{F}_{2^n}$ can be represented as n-dimensional binary vectors. This representation simplifies mathematical operations and manipulations, making it easier to work with cryptographic algorithms.

The Advanced Encryption Standard (AES) [25], also known as Rijndael, is a prime example of this. AES is a widely used symmetric-key block cipher, and it has a $(8, 8)$-function at its core. In AES, the field $\mathbb{F}_{2^8}$ is used, which corresponds to 8-bit binary representations. Each element in $\mathbb{F}_{2^8}$ can be thought of as an 8-bit binary vector.

This representation allows AES to perform bitwise operations and transformations on these binary vectors efficiently, making it a versatile and secure encryption algorithm. It demonstrates how the choice of representation can significantly impact the efficiency and effectiveness of cryptographic algorithms.

When a vectorial Boolean function is utilized as an S-box, it is obviously crucial to examine its resilience to different cryptanalytic attacks. Differential cryptanalysis [2] is one of the most effective methods of attacking block ciphers. It is about how differences in input a can affect the resultant difference in output b.

Derivative of a function F, denoted as D_aF, or the (n, n)-function D_aF, is a mathematical tool used to explain how a change in the input of a (n, n)-function F affects the resulting difference in the output. It helps analyze the relationship between differences in inputs and their corresponding differences in outputs. The (n, n)-function D_aF is defined as:

$$D_aF(x) = F(x + a) + F(a).$$

In essence, $D_aF(x)$ quantifies how a change in the input vector x by the direction a influences the change in the output of the function F. This concept is often used in the analysis of cryptographic functions to understand their behaviour when subjected to variations in input values.

In [62], the idea of differential uniformity of a function is proposed as a way to

quantify the function's contribution to the block cipher's resistance to differential cryptanalysis. The largest number of solutions $x \in \mathbb{F}_{2^n}$ to any equation of the form $D_aF(x) = b$ is known as the differential uniformity Δ_F of a (n, n)-function F, i.e. $F(x) + F(x + a) = b$ for $a, b \in \mathbb{F}_{2^n}$ with $a \neq 0$,,

$$\Delta_F = max_{a \in F_{2^n}^*, b \in \mathbb{F}_{2^m}} |\{x \in \mathbb{F}_{2^n} : F(a + x) + F(x) = b\}|.$$

If x_1 is a solution of the equation $F(a+x)+F(x) = b$, then $x_1 + a$ is also a solution, then Δ_F must be even for any F, and hence must be greater than or equivalent to 2. Indeed, almost perfect nonlinear (APN) functions are a class of (n, n)-functions where $\Delta_F = 2$. The parameter Δ_F represents the maximum differential uniformity of the function, and when it equals 2, it signifies that the function has optimal resistance to differential cryptanalysis.

A quadratic function F is APN if and only if the following equation holds for every $A \in \mathbb{F}_{2^n}^*$, where $\mathbb{F}_{2^n}^*$ represents the set of non-zero elements in $\mathbb{F}_{2^n}^*$ and the equation has exactly two solutions:

$$F(X + A) + F(X) + F(A) + F(0) = 0.$$

In other words, for each non-zero element A in the finite field $\mathbb{F}_{2^n}$, the equation above should have precisely two solutions for the variable X.

The Walsh transform is a useful tool for examining any (n, m)-function F. The Walsh transform of $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ is defined as

$$W_F(a, b) = \sum_{x \in \mathbb{F}_2^n} (-1)^{Tr_n(ax+bF(x))}$$

for $a, b \in \mathbb{F}_2^n$.

Nonlinearity in (n, m)-functions and the relationship with almost bent (AB) functions and bent functions is crucial in the context of cryptographic function analysis. Nonlinearity measures how closely an (n, m)-function F resembles a linear transformation. It quantifies the deviation from linearity. The nonlinearity of an (n, m)-function F can be expressed using the formula:

$$2^{n-1} - \frac{1}{2} max_{a \in \mathbb{F}_{2^n}^*, b \in \mathbb{F}_{2^m}} |W_F(a, b)|.$$

For (n, n)-functions, the nonlinearity is limited from above by $2^{n-1} - 2^{\frac{n-1}{2}}$ [23]. Functions achieving this upper bound are known as almost bent (AB) functions.

AB functions are a subset of (n, n)-functions that attain the highest possible nonlinearity for odd values of n. They offer the highest level of resistance to linear cryptanalysis. AB functions provide strong resistance to both differential and linear cryptanalysis. AB functions have certain limitations, such as challenges in combining them with vectorial functions to achieve high algebraic degrees.

Bent functions are defined for even values of n and have nonlinearity equal to $2^{n-1} - 2^{\frac{n}{2}}$. They are conjectured to have the highest possible nonlinearity for even values of n. Bent functions are highly resistant to various cryptanalysis techniques.

So, the nonlinearity of cryptographic functions, particularly (n, m)-functions, plays a critical role in their resistance to cryptanalysis. AB functions and bent functions are examples of functions that achieve optimal nonlinearity levels, and they offer strong resistance to both differential and linear cryptanalysis. However, they may have limitations [23] when it comes to achieving high algebraic degrees, which is also an important consideration in the design of cryptographic algorithms.

2.1.4 Equivalence Relations on Vectorial Boolean Functions

In the study and classification of almost perfect nonlinear (APN) and almost bent (AB) functions in the literature, various equivalence relations are used to group functions that exhibit similar cryptographic properties. Two commonly used equivalence relations are CCZ-equivalence and EA-equivalence. Both CCZ-equivalence and EA-equivalence are valuable tools in the analysis and classification of cryptographic functions. They help researchers identify functions that can be used interchangeably in cryptographic algorithms while preserving desired security properties, such as resistance to differential attacks and nonlinearity. These equivalence relations simplify the study of function classes and aid in the design and evaluation of secure cryptographic systems. But for several significant families of functions, the two equivalence relations coincide. Most notably, two quadratic APN functions are CCZ-equivalent if and only if they are EA-equivalent [62]. Due to this fact, EA-equivalence may be

simpler to use in some situations. In the context of power functions, particularly in finite fields, it is indeed the case that CCZ-equivalence and cyclotomic equivalence are equivalent [63]. This is a notable property when dealing with power functions, and it simplifies the classification and analysis of such functions.

2.1.4.1 EA-equivalence

Affine equivalence and extended affine (EA) equivalence are important in the study of cryptographic functions, especially in the context of analyzing their properties and transformations.

Two functions $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ and $A_1 \circ F \circ A_2$, where $A_1, A_2 = \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ are affine permutations, are called affine equivalent if they can be transformed into each other through a composition of affine permutations. This means that they exhibit similar properties under affine transformations.

Affine equivalence is used to group functions that behave similarly under affine transformations, which are commonly encountered in cryptographic algorithms.

Two functions F and F', where F' is defined as $F' = A_1 \circ F \circ A_2 + A$, and A is an affine function, are called extended affine (EA)-equivalent if they can be transformed into each other through an affine transformation involving an additional affine function A. In other words, they are equivalent under a more general class of transformations that include both affine permutations and an additional affine term.

EA equivalence is a broader equivalence relation that encompasses affine equivalence. It allows for the consideration of more general transformations that may involve an extra affine component.

Notably, if the original function F is not affine, then the functions F and F' (related through EA equivalence) have the same algebraic degree. This means that their algebraic properties remain unchanged even after the additional affine transformation.

EA equivalence is useful in cryptographic analysis when dealing with functions that may not be strictly affine but can be transformed into each other through a combination of affine operations and an affine component. It provides a more flexible

framework for analyzing and classifying functions in various cryptographic contexts.

2.1.4.2 CCZ-equivalence

Carlet-Charpin-Zinoviev equivalence (CCZ-equivalence) is a valuable tool in the study and classification of cryptographic functions, and it offers a more general equivalence relation than extended affine (EA) equivalence.

Two functions $F, F' = \mathbb{F}_{2^n} \rightarrow \mathbb{F}_{2^n}$ are called CCZ-equivalent if their respective graphs, denoted as G_F and $G_{F'}$, are affine equivalent where $G_F = \{(x, F(x)) \mid x \in \mathbb{F}_{2^n}\}$ and $G_{F'} = \{(x, F'(x)) \mid x \in \mathbb{F}_{2^n}\}$. In other words, they exhibit similar properties under affine transformations of their input-output pairs.

CCZ-equivalence implies the existence of an affine automorphism $\mathcal{L} = (L_1, L_2)$ of $\mathbb{F}_{2^n} \times \mathbb{F}_{2^n}$ such that for any pair (x, y) in the graph G_F such that $y = F(x)$, the corresponding pair in the graph $G_{F'}$ satisfies $L_2(x, y) = F'(L_1(x, y))$.

Because $\mathcal{L}$ is an affine automorphism and thus a permutation, the function $L_1(x, F(x))$ must also be a permutation [12]. This means that it bijectively maps elements of $\mathbb{F}_{2^n}$ to itself.

CCZ-equivalence is more general than EA-equivalence [10]. While EA-equivalence considers transformations that involve affine permutations and an additional affine term, CCZ-equivalence encompasses a broader set of affine transformations that align the graphs of two functions under an affine automorphism.

Importantly, any permutation is CCZ-equivalent to its inverse [21], highlighting the generality and flexibility of CCZ-equivalence in capturing the properties of cryptographic functions.

2.1.4.3 Cyclotomic equivalence

Cyclotomic equivalence is used to classify two power functions $F(x) = x^d$ and $G(x) = x^e$ over $\mathbb{F}_{2^n}$ based on their exponents d and e, where d,e and n are positive integers.Two power functions are considered cyclotomic equivalent if specific

conditions related to their exponents are met.

Two power functions $F(x) = x^d$ and $G(x) = x^e$ over $\mathbb{F}_{2^n}$ are considered cyclotomic equivalent if one of the following conditions is satisfied:

- $d \equiv 2^k e \mod (2^n - 1)$ for some positive integer k.
- $d^{-1} \equiv 2^k e \mod (2^n - 1)$ for some positive integer k if $gcd(d, 2^n - 1) = 1$, where d^{-1} represents the multiplicative inverse of d modulo $2^n - 1$.

Cyclotomic equivalence offers the advantage of being easier to test than both extended affine (EA) and Carlet-Charpin-Zinoviev (CCZ) equivalence, which involve more complex transformations and conditions.

Cyclotomic equivalence relies on relationships between the exponents d and e modulo $2^n - 1$. These relationships help classify power functions that have similar properties and behavior when it comes to exponentiation.

The use of modular arithmetic and the consideration of the greatest common divisor (gcd) are central to determining cyclotomic equivalence.

In summary, cyclotomic equivalence provides a simpler and more direct way to classify power functions based on their exponents and how they relate modulo $2^n - 1$. It is particularly useful when analyzing and categorizing APN power functions in cryptographic contexts, where modular arithmetic plays a significant role.

2.1.5 APN Power Functions

Power functions were the first discovered APN functions since if we take $F(x) = x^d$, power function, then F is APN if and only if the derivative D_1F is a two-to-one mapping i.e. for any $a \neq 0$

$$D_aF(x) = (x + a)^d + x^d = a^d D_1F(x/a)$$

then D_aF is two-to-one mapping if and only if D_1F is two-to-one.

As mentioned before there are six known infinite families of APN monomials presented in Table 2.1.

Table 2.1: Known APN power functions x^d over $\mathbb{F}_2^n$.

Functions	Exponents d	Conditions	Degree	In
Gold	2^i+1	$gcd(i,n)=1$	2	[37], [53]
Kasami	$2^{2i}-2^i+1$	$gcd(i,n)=1$	$i+1$	[41], [43]
Welch	2^t+3	$n=2t+1$	3	[33]
Niho	$2^t+2^{\frac{t}{2}}-1$, t even $2^{\frac{3t+1}{2}}+2^t-1$, t odd	$n=2t+1$	$\frac{t+2}{2}$ $t+1$	[32]
Inverse	$2^{2t}-1$	$n=2t+1$	$2t$	[1], [53]
Dobbertin	$2^{4i}+2^{3i}+2^{2i}+2^i-1$	$n=5i$	$i+3$	[34]

The conjecture by Dobbertin [34] that the classification of monomial almost perfect nonlinear (APN) functions is comprehensive up to CCZ-equivalence suggests that any monomial APN function can be transformed, through CCZ-equivalence, into an instance belonging to one of the families of functions listed in Table 2.1. The power APN functions play an important role through the known APN functions. The Gold, Kasami, Welch, and Niho APN power functions are well-known in the field of cryptography for their cryptographic properties. These functions exhibit a special property related to their classical Walsh spectrum, depending on whether n is odd or even:

These functions are considered "AB," which stands for almost bent functions when n is odd. AB functions are a subset of APN functions and have the highest possible nonlinearity for odd values of n. In this case, the functions are known to have optimal resistance to differential cryptanalysis.

For even values of n, the Walsh coefficients of these functions take values from the set $\{0, \pm 2^{n/2}, \pm 2^{n+2/2}\}$ in their classical Walsh spectrum. Even though they are not strictly AB for even n, they still exhibit strong cryptographic properties and provide resistance to certain types of cryptanalysis.

When it comes to inverse function, it is differentially 4-uniform for n is even and for n is odd, it is APN. Since the algebraic degree of any AB function cannot be more than $(n+1)/2$, the inverse APN function which has the algebraic degree $n-1$ is not an AB function [21]. The inverse function's Walsh spectrum was determined in [46] by Lachaud and Wolfman. The inverse function has the best-known nonlinearity for n even since it contains all integers with $t = 0 \bmod 4$ in the range $-2^{n/2+1}, \ldots, 2^{n/2+1}$ if n is even.

Canteaut and Dobbertin independently discovered the final instance of APN power functions in 1999, and Dobbertin proved it in 2000 [35]. The Dobbertin family of power functions' nonlinearity and also the Walsh spectrum remain unknown until some observations and computational data on the differential spectrum of power functions x^d with an exponent of the form $d = \Sigma_{i=1}^{k-1} 2^{ni} - 1$ over $\mathbb{F}_2^{nk}$; since the exponents of both the inverse and the Dobbertin family are special cases of this form, are presented in [9].

2.1.6 Known Infinite Families of Non-power APN Functions

In this section, we will present an overview of the known infinite families of non-power APN (Almost Perfect Nonlinear) functions, as stated [26]. The exploration of APN and AB (Almost Bent) functions involved considerations of EA-equivalence and permutation inverses until the construction of non-power APN functions was achieved through CCZ-equivalence, as described in [16]. Notably, when employed alongside Gold power APN functions, it was revealed in [16] that CCZ-equivalence offers a broader scope than EA-equivalence combined with inverse transformations. Consequently, this led to the construction of the first classes of APN and AB functions that are EA-inequivalent to power functions. This research also challenged the previously held notion that all AB functions could be EA-equivalent to permutations, as suggested in [16], and refuted the conjecture put forth in [21]. Moreover, it was demonstrated in [14] and [10] that for Gold power functions, other quadratic APN polynomials, and APN polynomials CCZ-inequivalent to both quadratic and power functions, CCZ-equivalence can be more general than EA-equivalence when combined with inverse transformations.

In [36], a significant breakthrough was achieved with the discovery of the first instances of APN functions that exhibit CCZ-inequivalence in comparison to power functions. These instances took the form of binomials: $x^3 + wx^{528}$ over $\mathbb{F}_{2^{12}}$ and $x^3 + wx^{36}$ over $\mathbb{F}_{2^{10}}$. The inspiration for these discoveries stemmed from considering the combination of two Gold APN functions. Subsequently, the first infinite classes of APN binomials, which demonstrated CCZ-inequivalence to power functions, were established as a generalization of the polynomial $x^3 + wx^{528}$ presented in [12]. These

classes are designated as $F1$ and $F2$ and are seen in Table 2.2.

The family of hexanomials $F3$ was established by extending a construction technique originally introduced by J. Dillon and outlined in [36]. In this extended approach, the focus was on quadratic polynomials expressed as $F(x) = x(Ax^2 + Bx^q + Cx^{2q}) + x^2(Dx^q + Ex^{2q}) + Gx^{3q}$ over $\mathbb{F}_{2^{2m}}$, where $q = 2^m$. Through this extension, new instances of quadratic APN functions in 6 and 8 variables were generated, demonstrating their CCZ-inequivalence to power functions, as described in the original approach detailed in [6].

In [14], it was observed that functions in the form of $F + f$, where F represents an APN function and f is a Boolean function, exhibit a maximum differential uniformity of 4. This insight led to the creation of the $F4$ family, encompassing both APN and AB functions. It is important to note that among all families of APN functions, only the $F4$ family remains inequivalent to power functions across all values of n.

When considering linear functions denoted as L_1 and L_2, the construction involving expressions like $L_1(x^3) + L_2(x^9)$ led to the discovery of two additional infinite families of APN and AB functions, namely, $F5$ and $F6$ (refer to Table 2.2).

By employing the isotopic shift construction method, the family denoted as $F11$ in Table 2.2 was derived in [8]. The aim was to establish a broader equivalence relation than CCZ-equivalence by utilizing isotopic equivalence principles applied to planar functions. These functions are defined over the field $\mathbb{F}_{p^n}$, where p is a prime greater than 2, and are extended to vectorial functions. Interestingly, this endeavour led to the discovery of a novel method for constructing APN functions distinct from power functions.

A very effective method for the building of new APN functions is the so-called bivariate construction of APN functions, which was first described in [19]. This approach led to the introduction of the $F10$, $F12$, $F14$ and $F15$ infinite families of APN functions (see [8], [38], [59], [65]).

APN polynomials, which are CCZ-inequivalent to monomials, are all quadratic at this time. In dimension 6, there is just one known instance of an APN function that is neither quadratic nor power.

Table 2.2: Infinite families of quadratic APN polynomials over the finite field $\mathbb{F}_{2^n}$ that are well-known

Family	Functions d	Conditions	Source
$F1-F2$	$x^{2^s+1}+u^{2^k-1}x^{2^{ik}+2^{mk+s}}$	$n=pk$,$gcd(k,3)=gcd(s,3k)=1$, $p\in\{3,4\}$,$i=sk$ mod p,$m=p-i$,$n\geq 12$,u primitive in $\mathbb{F}_{2^n}^*$	[13]
$F3$	$sx^{q+1}+x^{2^i+1}+x^{q(2^i+1)}+cx^{2^iq+1}+c^qx^{2^i+q}$	$q=2^m, n=2m$, gcd$(i,m)=1, c\in\mathbb{F}_{2^n}, s\in\mathbb{F}_{2^n}/\mathbb{F}_q, X^{2^i+1}+cX^{2^i}+c^qX+1$ has no solution x s.t. $x^{q+1}=1$	[11]
$F4$	$x^3+a^{-1}Tr_1^n(a^3x^9)$	$a\neq 0$	[14]
$F5$	$x^3+a^{-1}Tr_3^n(a^3x^9+a^6x^{18})$	$3\|n, a\neq 0$	[15]
$F6$	$x^3+a^{-1}Tr_3^n(a^6x^{18}+a^{12}x^{36})$	$3\|n, a\neq 0$	[15]
$F7-F9$	$ux^{2^s+1}+u^{2^k}x^{2^{-k}+2^{k+s}}+vx^{2^{-k}+1}+wu^{2^k+1}x^{2^s+2^{k+s}}$	$n=3k, gcd(k,3)=gcd(s,3k)=1, v,w\in\mathbb{F}_{2^k}, vw\neq 1, 3\|(k+s)$,u primitive in $\mathbb{F}_{2^n}^*$	[4, 5]
$F10$	$(x+x^{2^m})^{2^k+1}+u'(ux+u^{2^m}x^{2^m})^{(2^k+1)2^i}+u(x+x^{2^m})(ux+u^{2^m}x^{2^m})$	$n=2m, m\geq 2$ even, $gcd(k,m)=1$ and $i\geq 2$ even, u primitive in $\mathbb{F}_{2^n}^*, u'\in\mathbb{F}_{2^m}$ not a cube	[65]
$F11$	$a^2x^{2^{2m+1}+1}+b^2x^{2^{m+1}+1}+ax^{2^{2m}+2}+bx^{2^m+2}+(c^2+c)x^3$	$n=3m$,m odd,$L(x)=ax^{2^{2m}}+bx^{2^m}+cx$ satisfies the conditions of Lemma 8 of [8]	[8]
$F12$	$u(u^qx+x^qu)(x^q+x)+(u^qx+x^qu)^{2^{2i}+2^{3i}}+a(u^qx+x^qu)^{2^{2i}}(x^q+x)^{2^i}+b(x^q+x)^{2^i+1}$	$q=2^m, n=2m, gcd(i,m)=1, x^{2^i+1}+ax+b$ has no roots in $\mathbb{F}_{2^m}$	[59]
$F13$	$x^3+a(x^{2^i+1})^{2^k}+bx^{3\cdot 2^m}+c(x^{2^{i+m}+2^m})^{2^k}$	$n=2m=10, (a,b,c)=(\beta,0,0), i=3, k=2, \mathbb{F}_4^*=<\beta>$ or $n=2m, m$ odd,$3\nmid m, (a,b,c)=(\beta,\beta^2,1), \mathbb{F}_4^*=<\beta>, i\in\{m-2, m, 2m-1, (m-2)^{-1}$ mod $m\}$	[17]
$F14$	$u[(u^qx+x^qu)^{2^i+1}+(u^qx+x^qu)(x^q+x)^{2^i}+(x^q+x)^{2^i+1}]+(u^qx+x^qu)^{2^{2i}+1}+(u^qx+x^qu)^{2^{2i}}(x^q+x)+(x^q+x)^{2^{2i}+1}$	$q=2^m, n=2m, gcd(3i,m)=1$, u primitive in $\mathbb{F}_{2^n}^*$	[38]
$F15$	$u[(u^qx+x^qu)^{2^i+1}+(u^qx+x^qu)(x^q+x)^{2^i}+(x^q+x)^{2^i+1}]+(u^qx+x^qu)^{2^{3i}}(x^q+x)+(u^qx+x^qu)(x^q+x)^{2^{3i}}$	$q=2^m$,m odd, $n=2m, gcd(3i,m)=1$, u primitive in $\mathbb{F}_{2^n}^*$	[38]

$$x^3 + a^{17}(x^{17} + x^{18} + x^{20} + x^{24}) + Tr_2(x^{21}) + Tr_3(a^{18}x^9) +$$
$$a^{14}Tr_6(a^{52}x^3 + a^6x^5 + a^{19}x^7 + a^{28}x^{11} + a^2x^{13})$$

For further information on APN function constructions, take a look at [7, 18, 20].

2.2 Cyclotomic Fields

For a given set S, we denote its size as $\#S$, and we define $S^\star$ as S without the element 0. The magnitude of a complex number z belonging to the set of complex numbers is represented as $|z|$. We denote a field with q elements as $\mathbb{F}_q$, where q is defined as $q = p^n$ with n as a positive integer and p as an odd prime. The Trace function for an element α in $\mathbb{F}_q$ over $\mathbb{F}_p$ is introduced as $\mathrm{Tr}^m(\alpha) = \alpha + \alpha^p + \alpha^{p^2} + \cdots + \alpha^{p^{n-1}}$.

We introduce the terms *non-squares* and *squares* to classify elements in the field $\mathbb{F}_p^*$, which we denote as NSQ and SQ respectively. The *quadratic character* of $\mathbb{F}_p^*$ is denoted as η_0, and for simplicity, we denote p^* as $\eta_0(-1)p$.

We explore the creation of a cyclotomic field denoted as $\mathbb{Q}(\xi_p)$, obtained by extending the rational field $\mathbb{Q}$ with the inclusion of the complex primitive p-th root of unity ξ_p. This field $\mathbb{Q}(\xi_p)$ has the splitting field of the polynomial $x^p - 1$, a Galois extension of degree $p - 1$. The field basis for this extension within the cyclotomic field $\mathbb{Q}(\xi_p)$ is this subset $\{1, \xi_p, \xi_p^2, \ldots, \xi_p^{p-1}\}$. The Galois group $\mathrm{Gal}(\mathbb{Q}(\xi_p)/\mathbb{Q})$ is described as the set $\{\sigma_a : a \in F_p^*\}$, where σ_a represents the automorphism of $\mathbb{Q}(\xi_p)$ defined as $\sigma_a(\xi_p) = \xi_p^a$. Notably, the cyclotomic field $\mathbb{Q}(\xi_p)$ features a unique quadratic subfield, $\mathbb{Q}(\sqrt{p^*})$, with its Galois group denoted as $Gal(\mathbb{Q}(\sqrt{p^*})/\mathbb{Q}) = \{1, \sigma_\gamma\}$ for some $\gamma \in NSQ$.

For elements a in $\mathbb{F}_p^*$ and b in $\mathbb{F}_p$, it is evident that $\sigma_a(\xi_p^b) = \xi_p^{ab}$, and $\sigma_a(\sqrt{p^*}^m) = \eta_0^n(a)\sqrt{p^*}^m$. We report some useful results in the following lemmas:

Lemma 1. *[49]*

i.) $\sum_{a \in \mathbb{F}_p^\star} \eta_0(a) = 0$.

ii.) $\sum_{a \in \mathbb{F}_p^\star} \xi_p^{ab} = -1$ *for every* $b \in \mathbb{F}_p^\star$.

$iii.)$ $\sum_{a\in\mathbb{F}_p^\star}\eta_0(a)\xi_p^{ab}=\eta_0(b)\sqrt{p^*}=\begin{cases}\sqrt{p^*}, & \text{if } b\in SQ,\\ -\sqrt{p^*}, & \text{if } b\in NSQ.\end{cases}$

$iv.)$ $\sum_{a\in\mathbb{F}_p^\star}\eta_0(a)\xi_p^{a}=\sqrt{p^*}.$

$iv.)$ $\sum_{a\in\mathbb{F}_p}\xi_p^{a^2b}=\begin{cases}p, & \text{if } b=0,\\ \sqrt{p^*}, & \text{if } b\in SQ,\\ -\sqrt{p^*}, & \text{if } b\in NSQ.\end{cases}$

$v.)$ $\sum_{a\in\mathbb{F}_p}\xi_p^{ab}=\begin{cases}p, & \text{if } b=0,\\ 0, & \text{if } b\neq 0.\end{cases}$

Lemma 2. *[49, Theorem 5.15]* $G(\eta)=(-1)^{m-1}\sqrt{p^*}^{\,m}$.

Lemma 3. *[49, Theorem 5.33]*

$$\sum_{x\in\mathbb{F}_q}\xi_p^{\mathrm{Tr}^m(ax^2+bx)}=G(\eta)\eta(a)\xi_p^{-\mathrm{Tr}^m(\frac{b^2}{4a})}.$$

2.3 Linear Codes

When we consider a finite field $\mathbb{F}_p$ containing p elements, where p is an odd prime, we define an $[n,k,d]$ linear code denoted as C. This linear code C is characterized by being a k-dimensional linear subspace within $\mathbb{F}_p^n$, and it is established over the same finite field $\mathbb{F}_p$. Additionally, the Hamming distance of this code is specified as d. We define the Hamming weight of a codeword, denoted as $\mathbf{c}=(c_0,c_1,\ldots,c_{n-1})$, using the expression $\mathrm{wt}(\mathbf{c})=\#\{c_i\neq 0: 0\leq i\leq n-1\}$. Here, we count the number of non-zero components in the codeword.

The *dual code* of $\mathcal{C}$ is defined to be the set

$$\mathcal{C}^\perp=\{(u_1,\ldots,u_n)\in\mathbb{F}_p^n\colon\ u_1v_1+\cdots+u_nv_n=0 \text{ for all } (v_1,\ldots,v_n)\in\mathcal{C}\},$$

which is represented by $[n,n-k,d^\perp]$ over $\mathbb{F}_p$, where $d^\perp$ is the minimum Hamming distance of $\mathcal{C}^\perp$.

Now, let's consider A_w as the count of codewords within $\mathcal{C}$ that possess a weight of w. When we refer to the weight distribution of $\mathcal{C}$, we are essentially talking about the

sequence $(1, A_1, A_2, \ldots, A_n)$. A code $\mathcal{C}$ earns the label "t-weight" when the number of non-zero values in the sequence $(A_1, A_2, \ldots, A_n)$ equals t. This characterization plays a crucial role in error detection and correction capabilities. Its weight enumerator is represented by the polynomial $1 + A_1 y + \cdots + A_n y^n$, where A_ω signifies the count of non-zero codewords with a weight of ω in $\mathcal{C}$. Let $A_1^\perp$ represent the count of codewords with a weight of 1 in $\mathcal{C}^\perp$. The sequence $(1, A_1^\perp, A_2^\perp, \ldots, A_n^\perp)$ is the weight distribution of the dual code $\mathcal{C}^\perp$. The first four Pless power moments in [40, p.259] are given as

$$\begin{aligned}
\sum_{i=0}^{n} A_i &= p^k, \\
\sum_{i=0}^{n} iA_i &= n(p-1)p^{k-1} - A_1^\perp p^{k-1}, \\
\sum_{i=0}^{n} i^2 A_i &= p^{k-2}\left((p-1)n(qn-n+1) - (2pn-p-2n+2)A_1^\perp + 2A_2^\perp\right), \\
\sum_{i=0}^{n} i^3 A_i &= p^{k-3}((p-1)n(p^2n^2 - 2pn^2 + 3pn - p + n^2 - 3n + 2) \\
&\quad -(3p^2n^2 - 3p^2n - 6pn^2 + 12pn + p^2 - 6p + 3n^2 - 9n + 6)A_1^\perp \\
&\quad +6(pn - p - n + 2)A_2^\perp - 6A_3^\perp).
\end{aligned}$$

For the codes constructed in this paper, it's important to note that $A_1^\perp = 0$ because their defining sets do not include the element $(0, 0)$. Thus, one can easily verify their parameters by applying the first two Pless power moments:

$$\begin{aligned}
\sum_{i=0}^{n} A_i &= p^k, \\
\sum_{i=0}^{n} iA_i &= n(p-1)p^{k-1}.
\end{aligned}$$

2.4 Weakly regular bent functions

Consider a p-ary function $f : \mathbb{F}_q \longrightarrow \mathbb{F}_p$, where $\mathbb{F}_q$ is a finite field with q elements, and $\mathbb{F}_p$ is another finite field with p elements. The *Walsh transform* of f is a complex-valued function defined as

$$\mathcal{W}_f(\beta) = \sum_{x \in \mathbb{F}_q} \xi_p{}^{f(x) - \mathrm{Tr}^m(\beta x)}, \quad \beta \in \mathbb{F}_q.$$

A function $f : \mathbb{F}_q \longrightarrow \mathbb{F}_p$ is termed *bent* if $|\mathcal{W}_f(\beta)|^2 = p^n$ for every $\beta \in \mathbb{F}_q$ (see [54] for Boolean bent and [45] for p-ary bent).

A bent function f is referred to as *weakly regular* if it satisfies the condition: $\mathcal{W}_f(\beta) = up^{\frac{n}{2}}\xi_p^{f^\star(\beta)}$, where $u \in \{\pm 1, \pm i\}$, and $f^\star$ is a p-ary function defined over $\mathbb{F}_q$. If this condition is not met, f is classified as *non-weakly regular*. Specifically, when $u = 1$, the function f is termed *regular*. In this context, $f^\star$ is referred to as *the dual* of f, and it is also a weakly regular bent function if f is.

The following lemma proves to be highly valuable for computing the Hamming weights of the proposed codes in Chapter 5 in this thesis.

Lemma 4. *[58] Let f be a weakly regular bent function. Then, for every $\beta \in \mathbb{F}_q$,*

$$\mathcal{W}_f(\beta) = \epsilon_f \sqrt{p^*}^m \xi_p^{f^\star(\beta)}$$

where $\epsilon_f \in \{\pm 1\}$ represents the sign of $\mathcal{W}_f$, and $f^\star$ denotes the dual of f.

Consider a weakly regular bent function $f : \mathbb{F}_q \to \mathbb{F}_p$ satisfying two essential homogeneous conditions:

- $f(0) = 0$, and
- $f(ax) = a^{k_f} f(x)$ for all $x \in \mathbb{F}_q$ and $a \in \mathbb{F}_p^\star$, where k_f is an even positive integer with $\gcd(k_f - 1, p - 1) = 1$.

The collection of such weakly regular bent functions is denoted as $\mathcal{RF}$. In the context of this thesis, we will utilize the following result from [58] in Chapter 5:

Lemma 5. *[58] For any function $f \in \mathcal{RF}$, we have $f^\star(0) = 0$ and $f^\star(a\beta) = a^{l_f} f^\star(\beta)$ for all $a \in \mathbb{F}_p^\star$ and $\beta \in \mathbb{F}_q$, where l_f is a positive even integer with $\gcd(l_f - 1, p - 1) = 1$.*

The following lemma is used to determine the length of linear codes.

Lemma 6. *[58] Let $f : \mathbb{F}_q \to \mathbb{F}_p$ be an unbalanced function with $\mathcal{W}_f(0) = \epsilon_f \sqrt{p^*}^m$, where $\epsilon_f = \pm 1$ is the sign of $\mathcal{W}_f$. For $j \in \mathbb{F}_p$, define $\mathcal{N}_f(j) = \#\{x \in \mathbb{F}_q : f(x) = j\}$.*

- *If m is even, then*

$$\mathcal{N}_f(j) = \begin{cases} p^{m-1} + \epsilon_f \eta_0(-1)(p-1)\sqrt{p^*}^{m-2}, & \text{if } j = 0, \\ p^{m-1} - \epsilon_f \eta_0(-1)\sqrt{p^*}^{m-2}, & \text{if } j \in \mathbb{F}_p^\star. \end{cases}$$

- *If m is odd, then*

$$\mathcal{N}_f(j) = \begin{cases} p^{m-1}, & \text{if } j = 0, \\ p^{m-1} + \epsilon_f \sqrt{p^*}^{m-1}, & \text{if } j \in SQ, \\ p^{m-1} - \epsilon_f \sqrt{p^*}^{m-1}, & \text{if } j \in NSQ. \end{cases}$$

The following lemma is used to find the weight distributions of linear codes.

Lemma 7. *[58] Let $f \in \mathcal{RF}$ with $\mathcal{W}_f(0) = \epsilon_f \sqrt{p^*}^m$ and f^* be the dual of f. For $j \in \mathbb{F}_p$, define $\mathcal{N}_{f^*}(j) = \#\{\beta \in \mathbb{F}_q : f^*(\beta) = j\}$.*

- *If m is even, then*

$$\mathcal{N}_{f^*}(j) = \begin{cases} p^{m-1} + \epsilon_f \eta_0(-1)(p-1)\sqrt{p^*}^{m-2}, & \text{if } j = 0, \\ p^{m-1} - \epsilon_f \eta_0(-1)\sqrt{p^*}^{m-2}, & \text{if } j \in \mathbb{F}_p^\star. \end{cases}$$

- *If m is odd, then*

$$\mathcal{N}_{f^*}(j) = \begin{cases} p^{m-1}, & \text{if } j = 0, \\ p^{m-1} + \epsilon_f \eta_0(-1)\sqrt{p^*}^{m-1}, & \text{if } j \in SQ, \\ p^{m-1} - \epsilon_f \eta_0(-1)\sqrt{p^*}^{m-1}, & \text{if } j \in NSQ. \end{cases}$$

CHAPTER 3

APN FUNCTIONS SPECIFIC TO CARLET'S BIVARIATE APN CONSTRUCTION

Let $n = 2m$ and the function F be defined as $F : (x, y) \in \mathbb{F}_{2^n} \to (B(x, y), G(x, y)) \in \mathbb{F}_{2^m} \times \mathbb{F}_{2^m}$. In [19], Carlet considered the simplest Maiorana-McFarland function $B(x, y) = xy$ and gave three conditions for F to be APN:

i. The function $x \to G(x, y)$ is APN for any fixed y.

ii. The function $y \to G(x, y)$ is APN for any fixed x.

iii. The function $G(x, bx + c)$ is APN for any b and c.

Since G is quadratic, it can be assumed that $c = 0$. Then the following APN class is revealed.

Theorem 1 (Carlet [19]). *Let $n = 2m$; let i, j be such that $gcd(i - j, m) = 1$ and let $s, t \neq 0$, u and v in $\mathbb{F}_{2^m}$. Set $G(x, y) = sx^{2^i+2^j} + ux^{2^i}y^{2^j} + vx^{2^j}y^{2^i} + ty^{2^i+2^j}$. Then $F(x, y) = (xy, G(x, y))$ is APN if and only if the polynomial $G(X, 1) = sX^{2^i+2^j} + uX^{2^i} + vX^{2^j} + t$ has no zero in $\mathbb{F}_{2^m}$.*

This method was also used later by Zhou–Pott [65] and Taniguchi [59] to construct new APN classes. We record their results below.

Theorem 2 (Zhou–Pott [65]). *Let $n = 2m$, m even, and let i be such that $gcd(i, m) = 1$. Set $G(x, y) = x^{2^i+1} + \alpha y^{2^j(2^i+1)}$. Then $F(x, y) = (xy, G(x, y))$ is APN if and only if $\alpha \in \{u^{2^i+1}(t^{2^i} + t)^{1-2^j} : u, t \in \mathbb{F}_{2^m}\}$. In particular if j is even, then F is APN if and only if α is not a cube.*

Theorem 3 (Taniguchi [59]). *Let $n = 2m$ and i be an integer such that $gcd(i, m) = 1$. Set $G(x, y) = x^{2^{2i}+2^{3i}} + ax^{2^{2i}}y^{2^i} + by^{2^i+1}$ with $a \in 0, 1$. Then $F(x, y) = (xy, G(x, y))$ is APN if and only if $X^{2^i+1} + aX + b$ has no zero in $\mathbb{F}_{2^m}$.*

3.1 Three Cases of APN Functions From Carlet's Construction

Considering the following theorem and lemma in [18], we get new results on bivariate APN functions.

Theorem 4 ([18]). *Let*

$$F(x, y) = (xy, a(x^{2^i+1})^{2^k} + b(x^{2^i}y)^{2^h} + c(xy^{2^i})^{2^r} + d(y^{2^i+1})^{2^s})$$

be an APN function over $\mathbb{F}_{2^{2m}}$. Then, F is EA-equivalent to one of the following functions
$F_1(x, y) = (xy, x^{2^i+1} + x^{2^{i+h}}y^{2^h} + b'x^{2^k}y^{2^{i+k}} + c'y^{2^{i+r}+2^r})$,
$F_2(x, y) = (xy, x^{2^i+1} + x^{2^k}y^{2^{i+k}} + c'y^{2^{i+r}+2^r})$,
$F_3(x, y) = (xy, x^{2^i+1} + c'y^{2^{i+r}+2^r})$, *with $c' \neq 0$.*

The following lemma gives the requirement for the family F_1 given in Theorem 4 to be APN, when $h = m/2$, $k, r = 0$.

Lemma 8. *[18, Lemma 6.1] Let $n = 2m$ with $m > 2$ even. Let i coprime with m. If $F(x, y) = (xy, x^{2^i+1} + x^{2^{i+m/2}}y^{2^{m/2}} + bxy^{2^i} + cy^{2^i+1})$ is APN, then $cX^{2^i+1} + bX^{2^i} + X^{2^{m/2}} + 1$ has no zero in $\mathbb{F}_{2^m}$.*

We now give a general result for the family F_1 of functions. It is assumed in Lemma 9 that h, k, r, m are integers such that they divide each other in any order ending with m. It is assumed in the lemma without loss of generality that h divides k, k divides r, r divides m. But it holds under any order of divisibility condition.

Lemma 9. *Let h, k, r, m be positive integers such that h divides k, k divides r, r divides m. Let $n = 2m$ with $m > 2$ even. Let i coprime with m. If $F_1(x, y) = (xy, x^{2^i+1} + x^{2^{i+h}}y^{2^h} + bx^{2^k}y^{2^{i+k}} + cy^{2^{i+r}+2^r})$ is APN, then $cX^{2^{i+1}+2^r} + bX^{2^{i+k}} + X^{2^h} + 1$ has no zero in $\mathbb{F}_{2^m}$.*

Proof. Denote $F(x,y) = (xy, G(x,y))$ with $G(x,y) = x^{2^i+1} + x^{2^{i+h}}y^{2^h} + bx^{2^k}y^{2^{i+k}} + cy^{2^{i+r}+2^r}$. Since F_1 is APN, G satisfies the following conditions

i. The function $x \to G(x,y)$ is APN for any fixed y;

ii. The function $y \to G(x,y)$ is APN for any fixed x;

iii. The function $G(x, \beta x + \gamma)$ is APN for any β and γ.

The conditions (i) and (ii) are clearly satisfied. We deal with the third one. We can consider $\gamma = 0$ since G is quadratic. We have $G(x,\beta x) = x^{2^i+1} + \beta^{2^h}x^{2^{i+h}}x^{2^h} + b'\beta^{2^{i+k}}x^{2^k}x^{2^{i+k}} + c'\beta^{2^{i+r}+2^r}x^{2^{i+r}+2^r}$. If we take $x \in \mathbb{F}_{2^h}$ with h dividing k, k dividing r, r dividing m, then we have the function $F'(x) = (1+\beta^{2^h}+b\beta^{2^{i+k}}+c\beta^{2^{i+r}+2^r})x^{2^{i+1}}$. If there exist β such that $1 + \beta^{2^h} + b\beta^{2^{i+k}} + c\beta^{2^{i+r}+2^r} = 0$, then for any $a \in \mathbb{F}_{2^h}^*$ and for any $x \in \mathbb{F}_{2^h}$, $F'(x) + F'(x+a) = 0$ imply that $G(x,\beta x)$ is not APN. □

A similar no-root condition can be proven for a sub-class of the family F_1, which we give in the following theorem.

Theorem 5. *Let $n = 2m$ with $m > 2$ even. Let $i < m/2$ coprime with m, let $h = m/4$, $k = 2m/4$, $r = 3m/4$. Let $A = X^{2^h}$, $B = bX^{2^{i+k}}$, $C = cX^{2^{i+r}+2^r}$ be monomials for some $b, c \in \mathbb{F}_{2^m}$ and the determinant be a polynomial*

$$D(X) = \begin{vmatrix} 1 & A & B & C \\ C^{2^{m/4}} & 1 & A^{2^{m/4}} & B^{2^{m/4}} \\ B^{2^{m/2}} & C^{2^{m/2}} & 1 & A^{2^{m/2}} \\ A^{2^{3m/4}} & B^{2^{3m/4}} & C^{2^{3m/4}} & 1 \end{vmatrix} \in \mathbb{F}_{2^m}[X].$$

Then, $F_1(x,y) = (xy, x^{2^i+1} + x^{2^{i+h}}y^{2^h} + bx^{2^k}y^{2^{i+k}} + cy^{2^{i+r}+2^r})$ is APN if and only if $D(X) \in \mathbb{F}_{2^m}[X]$ has no zero.

Proof. We set $G(x,y) = x^{2^i+1} + x^{2^{i+h}}y^{2^h} + bx^{2^k}y^{2^{i+k}} + cy^{2^{i+r}+2^r}$, then $G(x,\beta x) = x^{2^i+1} + \beta^{2^h}x^{2^{i+h}}x^{2^h} + b\beta^{2^{i+k}}x^{2^k}x^{2^{i+k}} + c\beta^{2^{i+r}+2^r}x^{2^{i+r}+2^r}$. $G(x,\beta x) = L_\beta(x^{2^i+1})$ for $L_\beta(x) = x + \beta^{2^h}x^{2^h} + b\beta^{2^{i+k}}x^{2^k} + c\beta^{2^{i+r}+2^r}x^{2^r}$. Therefore, F_1 is APN if and only if L_β is a permutation of $\mathbb{F}_{2^m}$. We have $h = m/4$, $k = 2m/4$, $r = 3m/4$. Then

$L_\beta(x) = x + Ax^{2^{m/4}} + Bx^{2^{2m/4}} + Cx^{2^{3m/4}}$ for $A = \beta^{2^h}$, $B = b\beta^{2^{i+k}}$, $C = c\beta^{2^{i+r}+2^r}$. L_β is a permutation if and only if

$$D(\beta) = \begin{vmatrix} 1 & A & B & C \\ C^{2^{m/4}} & 1 & A^{2^{m/4}} & B^{2^{m/4}} \\ B^{2^{m/2}} & C^{2^{m/2}} & 1 & A^{2^{m/2}} \\ A^{2^{3m/4}} & B^{2^{3m/4}} & C^{2^{3m/4}} & 1 \end{vmatrix}$$

is nonzero. L_β is a permutation for any β if and only if the polynomial $D(X) \in \mathbb{F}_{2^m}[X]$ has no zero. □

We extend the necessary condition given [18] for a function F_2. Similar to Lemma 9, we assume in Lemma 10 that h, k, r, m are integers such that they divide each other in any order ending with m.

Lemma 10. *Let h, k, r, m be integers such that they divide each other in any order ending with m with $m > 2$. Let $n = 2m$ and i be co-prime with m. If $F_2(x, y) = (xy, x^{2^i+1} + x^{2^k}y^{2^{i+k}} + cy^{2^{i+r}+2^r})$ is APN, then $cX^{2^{i+1}+2^r} + X^{2^{i+k}} + 1$ has no zero in $\mathbb{F}_{2^m}$.*

Proof. We have $F(x, y) = (xy, G(x, y))$ with $G(x, y) = x^{2^i+1} + x^{2^k}y^{2^{i+k}} + cy^{2^{i+r}+2^r}$. As in the extension of the previous lemma, we dealt with the third condition. $G(x, \beta x) = x^{2^i+1} + \beta^{2^{i+k}}x^{2^k}x^{2^{i+k}} + c\beta^{2^{i+r}+2^r}x^{2^{i+r}+2^r}$. If we take $x \in \mathbb{F}_{2^k}$, since k divides r and r divides m, we have the function $F'(x) = (1 + \beta^{2^{i+k}} + c\beta^{2^{i+r}+2^r})x^{2^{i+1}}$. If there exist β such that $1 + \beta^{2^{i+k}} + c\beta^{2^{i+r}+2^r} = 0$, then for any $a \in \mathbb{F}_{2^k}^*$ and for any $x \in \mathbb{F}_{2^k}$ $F'(x) + F'(x + a) = 0$ imply that $G(x, \beta x)$ is not APN. □

A no-root condition for a sub-class of the family F_2 is presented in the following theorem.

Theorem 6. *Let $n = 2m$ with $m > 2$ even. Let $i < m/2$ coprime with m and $k = m/3$, $r = 2m/3$. Let $A = x^{2^{i+k}}$, $B = cX^{2^{i+r}+2^r}$ be monomials for some $c \in \mathbb{F}_{2^m}$ and the determinant $D(X)$ be a polynomial*

$$D(X) = \begin{vmatrix} 1 & A & B \\ B^{2^{m/3}} & 1 & A^{2^{m/3}} \\ A^{2^{2m/3}} & B^{2^{2m/3}} & 1 \end{vmatrix} \in \mathbb{F}_{2^m}[X].$$

Then, $F_2(x,y) = (xy, x^{2^i+1} + x^{2^k}y^{2^{i+k}} + cy^{2^{i+r}+2^r})$ *is APN if and only if* $D(X)$ *has no zero in* $\mathbb{F}_{2^m}[X]$.

Proof. We set $G(x,y) = x^{2^i+1} + x^{2^k}y^{2^{i+k}} + cy^{2^{i+r}+2^r}$ then $G(x,\beta x) = x^{2^i+1} + \beta^{2^{i+k}}x^{2^k}x^{2^{i+k}} + c\beta^{2^{i+r}+2^r}x^{2^{i+r}+2^r}$. $G(x,\beta x) = L_\beta(x^{2^i+1})$ for $L_\beta(x) = x + \beta^{2^{i+k}}x^{2^k} + c\beta^{2^{i+r}+2^r}x^{2^r}$. F_2 is APN if and only if L_β is a permutation of $\mathbb{F}_{2^m}$. We note that L_β is a linearized polynomial. We have $k = m/3$, $r = 2m/3$. Then, $L_\beta(x) = x + Ax^{2^{m/3}} + Bx^{2^{2m/3}}$ for $A = \beta^{2^{i+k}}$, $B = c\beta^{2^{i+r}+2^r}$. L_β is a permutation for any β if and only if

$$\begin{vmatrix} 1 & A & B \\ B^{2^{m/3}} & 1 & A^{2^{m/3}} \\ A^{2^{2m/3}} & B^{2^{2m/3}} & 1 \end{vmatrix}$$

is nonzero. So L_β is a permutation for any β if and only if the polynomial $D(X) \in \mathbb{F}_{2^m}[X]$ has no zero. □

In the remaining part of this section, we deal with the family F_3, and present sufficient condition in Lemma 11 and a necessary-sufficient condition in Theorem 7 for some of its subclasses.

Lemma 11. *Let* $n = 2m$ *with* $m > 2$ *even. Let* i *coprime with* m *and* r *divides* m. *If*

$$F_3(x,y) = (xy, x^{2^i+1} + cy^{2^{i+r}+2^r})$$

is APN, then $cX^{2^{i+1}+2^r} + 1$ *has no zero in* $\mathbb{F}_{2^m}$.

Proof. We set $F(x,y) = (xy, G(x,y))$ with $G(x,y) = x^{2^i+1} + cy^{2^{i+r}+2^r}$. As in the extension of previous lemmas we dealt with the third condition. $G(x,\beta x) = x^{2^i+1} + c\beta^{2^{i+r}+2^r}x^{2^{i+r}+2^r}$.
If we take $x \in \mathbb{F}_{2^r}$, and if r divides m then we have the function $F'(x) = (1 + c\beta^{2^{i+r}+2^r})x^{2^i+1}$. If there exist β such that $1 + c\beta^{2^{i+r}+2^r} = 0$, then for any $a \in \mathbb{F}_{2^r}^*$ and for any $x \in \mathbb{F}_{2^r}$, $F'(x) + F'(x+a) = 0$ imply that $G(x,\beta x)$ is not APN. □

Theorem 7. *Let* $n = 2m$ *with* $m > 2$ *even. Let* $i < m/2$ *coprime with* m *and* $r = m/2$. *Then,*

$$F_3(x,y) = (xy, x^{2^i+1} + cy^{2^{i+r}+2^r})$$

is APN if and only if

$$1 + (cX^{2^{i+r}+2^r})^{2^{m/2+1}}$$

has no zero in $\mathbb{F}_{2^m}$.

Proof. We set $G(x, y) = x^{2^i+1} + cy^{2^{i+r}+2^r}$ then $G(x, \beta x) = x^{2^i+1} + c\beta^{2^{i+r}+2^r} x^{2^{i+r}+2^r}$ and $G(x, \beta x) = L_\beta(x^{2^i+1})$ for $L_\beta(x) = x + c\beta^{2^{i+r}+2^r} x^{2^r}$. F_3 is APN if and only if L_β is a permutation of $\mathbb{F}_{2^m}$. $L_\beta(x) = x + Ax^{2^{m/2}}$, $A = c\beta^{2^{i+r}+2^r}$. L_β is a permutation if and only if

$$\begin{vmatrix} 1 & A \\ A^{2^{m/2}} & 1 \end{vmatrix}$$

is nonzero.

So L_β is a permutation for any β if and only if

$$1 + (cX^{2^{i+r}+2^r})^{2^{m/2+1}}$$

has no zero in $\mathbb{F}_{2^m}$. □

3.2 Biprojective Almost Perfect Nonlinear Functions

Let $q = 2^k$ and $m \in \mathbb{Z}^+$. A polynomial of type $X^{q+1} + aX^q + bX + c \in \mathbb{F}_{2^m}[X]$ is called projective, and a polynomial of type $ax^{q+1} + bx^q y + cxy^q + dy^{q+1} \in \mathbb{F}_{2^m}[x, y]$ is called bivariate projective (or bi-projective) polynomial of degree $q + 1$. Using bi-projective polynomials is an extension method of the construction methods given in Section 3.1.

A general representation of the bivariate construction with two biprojective polynomials over $\mathbb{F}_{2^m}$ is given by

$$F(x, y) = (Ax^{2^i+1} + Bx^{2^i} y + Cxy^{2^i} + Dy^{2^i+1}, ax^{2^j+1} + bx^{2^j} y + cxy^{2^j} + dy^{2^j+1}).$$

Then we consider Theorem 7.1 in [18] as follows;

Theorem 8. *[18, Theorem 7.1] Let m be positive even integer, $n = 2m$, $i, j \leq m/2$ and*

$$F(x, y) = (Ax^{2^i+1} + Bx^{2^i} y + Cxy^{2^i} + Dy^{2^i+1}, ax^{2^j+1} + bx^{2^j} y + cxy^{2^j} + dy^{2^j+1})$$

be APN over $\mathbb{F}_{2^n}$. Then F is EA-equivalent to one of the following function:

1. $f_1(x,y) = (x^{2^i+1} + x^{2^i}y + Axy^{2^i} + By^{2^i+1}, x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1})$
2. $f_2(x,y) = (x^{2^i+1} + xy^{2^i} + Ay^{2^i}, x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1})$
3. $f_3(x,y) = (x^{2^i+1} + Ay^{2^i+1}, x^{2^j+1} + x^{2^j}y + axy^{2^j} + by^{2^j+1})$
4. $f_4(x,y) = (x^{2^i+1} + Ay^{2^i+1}, x^{2^j+1} + axy^{2^j} + by^{2^j+1})$, *where* $a \in \{0,1\}$.

In the following subsections, we look for the conditions of Families f_1, f_2, f_3 and f_4 by using Carlet's method that is used in Section 3.1. But, a general result is not recorded here. It would be a future work to extend the observations in the following subsections.

3.2.1 Family f_1

We now consider the function $f_1(x,y) = (x^{2^i+1} + x^{2^i}y + Axy^{2^i} + By^{2^i+1}, x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1})$ and prove new requirements in order to say that f_1 is APN by using similar arguments as in Section 3.1.

We first consider the case $i = 0$. If $i = 0$ is substituted, we get

$$f_1(x,y) = (x^2 + xy + Axy + By^2, x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1}).$$

We define $B_{1,0}(x,y) := x^2 + xy + Axy + By^2$ and $G_{1,0}(x,y) := x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1}$. We know that the function f_1 is APN if the system of equations

$$B_{1,0}(x,y) + B_{1,0}(x+a, y+b) = c$$

$$G_{1,0}(x,y) + G_{1,0}(x+a, y+b) = d$$

has 0 or 2 solutions for every nonzero $a, b \in \mathbb{F}_{2^m}^*$, and for every $c, d \in \mathbb{F}_{2^m}$. As $B_{1,0}(x,y) = x^2 + xy + Axy + By^2$, we get $bx + ay + Abx + Aay = c$ by replacing c with $c + ab + Aab + Bb^2 + a^2$ in the equation $B_{1,0}(x,y) + B_{1,0}(x+a, y+b) = c$. Then, we need to check the number of solutions in the equation

$$G_{1,0}(x, \frac{c + b(A+1)x}{a(A+1)}) + G_{1,0}(x+a, \frac{c + b(A+1)x}{a(A+1)} + b) = d,$$

for $a(A+1) \neq 0$. Changing c into $a(A+1)c$ and x into ax and defining $G_{a,b,c}(x) := G_{1,0}(ax, bx+c)$, we get that the function f_1 is APN if and only if $G_{a,b,c}$ is an APN function. So, we have the same result with Carlet's construction [19] of the case $B(x,y) = xy$, which is discussed in the previous section.

From this observation above, we have the following corollary.

Corollary 1. Let

$$f_1(x,y) = (x^2 + xy + Axy + By^2, x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1})$$

and define $G_{a,b,c}(x) := G_{1,0}(ax, bx+c)$. For $a(A+1) \neq 0$, the function f_1 is APN if and only if $G_{a,b,c}$ is an APN function.

For $i = 1$, we have

$$f_1(x,y) = (x^3 + x^2y + Axy^2 + By^3, x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1}),$$

and $B_{1,1}(x,y) := x^3+x^2y+Axy^2+By^3$, $G_{1,1}(x,y) := x^{2^j+1}+ax^{2^j}y+bxy^{2^j}+cy^{2^j+1}$. Dealing with the equation $B_{1,1}(x,y) + B_{1,1}(x+a, y+b) = c$, by changing c into $c+a^3+a^2b+Aab^2+bB^3$, we have $(a+b)x^2+(a^2+Ab^2)x+(Aa+Bb)y^2+(a^2+Bb^2)y = c$. Taking s, $u \neq 0$, v and t instead of $a+b$, a^2+Ab^2, $Aa+Bb$ and a^2+Bb^2 respectively, we get $sx^2+ux+vy^2+ty = c$. Then if we take $d = c+vy^2+ty$, we have $sx^2+ux+d = 0$. Taking $y = \frac{sx}{u}$ and $k = \frac{sd}{u^2}$, we have the formula $y^2+y+k = 0$. If m is odd, this formula has a root

$$y_1 = \sum_{j\in J} k^{2^j} = \sum_{i\in I} k^{2^i},$$

where $J = \{0, 2, 4, \ldots, m-1\}$ and $I = \{1, 3, 5, \ldots, m-2\}$.
For $m = 3$, $I = \{1\}$ and $J = \{0, 2\}$ we have $y_1 = k^2$. So the roots of the formula $(a+b)x^2 + (a^2+Ab^2)x + (Aa+Bb)y^2 + (a^2+Bb^2)y = c$ are $x_1 = \frac{a+b}{a^2+Ab^2}(\frac{c+(Aa+Bb)y^2+(a^2+Bb^2)y}{a^2+Ab^2})^2$ and $x_2 = \frac{a^2+Ab^2}{a+b}[(\frac{(a+b)(c+(Aa+Bb)y^2+(a^2+Bb^2)y}{(a^2+Ab^2)^2})^2 + 1]$ for $a^2+Ab^2 \neq 0$.

Now, we consider the number of solutions in the equation $G_{1,1}(x,y)+G_{1,1}(x+a, y+b) = d$ by taking $A = B = 1$. Then

$$G_{1,1}((a+b)(c+y^2+y)^2, (a+b)y)+G_{1,1}((a+b)(c+y^2+y)^2+a_0, (a+b)y+b_0) = d$$

for every nonzero $a_0, b_0 \in \mathbb{F}_2^m$ must has no solution or one solution for the root x_1 and one solution for the root x_2. If $G_{1,1}$ has two solutions for the root x_1, then it has two solutions for the other root x_2. This contradicts the fact that $G_{1,1}$ is APN.

Then we have Corollary 2 below.

Corollary 2. Let

$$f_1(x,y) = (x^3 + x^2y + xy^2 + y^3, x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1})$$

. For $a^2 + b^2 \neq 0$ the function f_1 is APN if and only if

$$G_{1,1}((a+b)(c + y^2 + y)^{2}, (a+b)y) + G_{1,1}((a+b)(c + y^2 + y)^{2} + a_0, (a+b)y + b_0) = d$$

must has no solution or one solution for the root x_1 and one solution for the root x_2 for every nonzero $a_0, b_0 \in \mathbb{F}_2^m$.

3.2.2 Family f_2

In this section, we consider $f_2(x,y) = (x^{2^i+1} + xy^{2^i} + Ay^{2^i}, x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1})$. For $i = 0$, we set

$$f_2(x,y) = (B_{2,0}(x,y), G_{2,0}((x,y)),$$

where $B_{2,0}(x,y) = x^2 + xy + Ay$ and $G_{2,0}(x,y) = x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1}$. If we change c into $c + a^2 + ab + Ab^2$ in the equation $B_{2,0}(x,y) + B_{2,0}((x+a, y+b) = c$ and $x = \frac{c+ay}{b}$ in $G_{2,0}(x,y) + G_{2,0}(x+a, y+b) = d$ we get $G_{2,0}(c+ay, yb) + G_{2,0}(c + ay + a, yb + b) = d$. So actually the result is again Carlet's function.

Obtaining this result we have the corollary as follows.

Corollary 3. Let

$$f_2(x,y) = (x^2 + xy + Ay, x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1})$$

and define $G_{a,b,c}(x) := G_{2,0}(ax, bx + c)$. For $b \neq 0$, f_2 is APN if and only if $G_{a,b,c}$ is an APN function.

For $i = 1$, then we set $B_{2,1}(x,y) := x^3 + xy^2 + Ay^3$. Changing c into $c + a^3 + Ab^3$ in $B_{2,1}(x,y) + B_{2,1}(x+a, y+b) = c$, we have $ax^2 + (a^2 + b^2)x = c + Aby^2 + Ab^2y$.

If we take $d = c + Aby^2 + Ab^2y$, $y = \frac{ax}{a^2+b^2}$ and $k = \frac{ad}{(a^2+b^2)^2}$ for $(a^2 + b^2) \neq 0$, we have $y^2 + y + k = 0$. For $m = 3$, we have the root $y_1 = k^2$. Then the roots of $x^2a + a^2x + b^2x + Aby^2 + Ab^2y = c$ are $x_1 = a\frac{(c+Aby^2+Ab^2y)^2}{(a+b)^3}$ and the other one $x_2 = \frac{a^2+b^2}{a}[(\frac{ad}{(a^2+b^2)^2})^2 + 1]$. For $A = 1$,

$$G_{2,1}(a(c+by(ay+by+b)^2,(a+b)y)+G_{2,1}(a(c+by(ay+by+b))^2+(a+b)a,(a+b)y+b)=d$$

must have zero solution or one solution for the root x_1 and one solution for the root x_2 due to the same reason for f_1.

From this computations, the following corollary is given.

Corollary 4. Let

$$f_2(x,y) = (x^3 + xy^2 + y^2, x^{2^j+1} + ax^{2^j}y + bxy^{2^j} + cy^{2^j+1}).$$

For $a^2 + b^2 \neq 0$ and $m = 3$ the function f_2 is APN if and only if the equation

$$G_{2,1}(a(c+by(ay+by+b)^2,(a+b)y)+G_{2,1}(a(c+by(ay+by+b))^2+(a+b)a,(a+b)y+b)=d$$

must have zero solution or one solution for the root x_1 and one solution for the root x_2.

3.2.3 Family f_3

Now we consider the function $f_3(x,y) = (x^{2^i+1} + Ay^{2^i+1}, x^{2^j+1} + x^{2^j}y + axy^{2^j} + by^{2^j+1})$.

For $i = 0$, we set $B_{3,0}(x,y) := x^2 + Ay^2$. Dealing with the number of solutions for $B_{3,0}(x,y) + B_{3,0}(x+a,y+b) = c$, we have $c = a^2 + Ab^2$. So f_3 is APN if and only if $G_{3,0}(x,y) = x^{2^j+1} + x^{2^j}y + axy^{2^j} + by^{2^j+1}$ is APN.

Then we have the following corollary.

Corollary 5. Let

$$f_3(x,y) = (x^2 + Ay^2, x^{2^j+1} + x^{2^j}y + axy^{2^j} + by^{2^j+1}).$$

f_3 is APN if and only if $G_{3,0}(x,y) = x^{2^j+1} + x^{2^j}y + axy^{2^j} + by^{2^j+1}$ is APN.

For $i = 1$, we define $B_{3,1}(x,y) = x^3 + Ay^3$. If we take $c = c + a^3 + Ab^3$, $d = c+Aby^2+Ab^2y$ and $m = 3$, the roots of the formula $B_{3,1}(x,y)+B_{3,1}(x+a,y+b) = c$

are $x_1 = a(\frac{d}{a^3})^2$ and $x_2 = a((\frac{d}{a^3})^2 + 1)$. Then $G_{3,1}(a(c + Aby^2 + Ab^2y)^2, a^3y) + G_{3,1}(a(c + Aby^2 + Ab^2y)^2 + a, a^3y + b) = d$ must has zero solution or one solution for the root x_1 and one solution for the root x_2 to be an APN function.

After these results, we can give the corollary below.

Corollary 6. Let

$$f_3(x, y) = (x^3 + Ay^3, x^{2^j+1} + x^{2^j}y + axy^{2^j} + by^{2^j+1}).$$

For $a \neq 0$, f_3 is APN if and only if the equation $G_{3,1}(a(c + Aby^2 + Ab^2y)^2, a^3y) + G_{3,1}(a(c + Aby^2 + Ab^2y)^2 + a, a^3y + b) = d$ must has zero solution or one solution for the root x_1 and one solution for the root x_2

We note that Family f_4 can be handled similarly to Family f_3.

CHAPTER 4

REPRESENTATIONS OF APN POWER FUNCTIONS

Power functions were the first known and simplest APN functions. These functions are of the form $F(x) = x^d$ for some natural number d. Until now, we know six infinite families of APN power functions, shown in table Table 2.1. In this table, the exponent d in the univariate form is shown in the second column and the conditions satisfied for $F(x) = x^d$ to be APN functions are shown in the third column. New representations of the Niho, Welch and Dobbertin functions are shown in [9] as the composition of the two power functions $x^i \circ x^{1/j}$, and it is proven that these representations are optimal. From this point of view, we study finding alternative representations of Dobbertin and Niho functions.

4.1 Dobbertin Power Function's Representations

In this section, it appears that we are considering a specific power function denoted as x^d, where the exponent d is given by:

$$d = 2^{4i} + 2^{3i} + 2^{2i} + 2^i - 1.$$

Additionally, it is mentioned that n is equal to $5i$.

The power function x^d is a monomial and is of particular interest in the study of almost perfect nonlinear (APN) functions in cryptography due to its cryptographic properties.

It has been demonstrated in [9] that the Dobbertin power function can be expressed as the composition of a cubic power function and the inverse of a quadratic power

function. This representation is optimal in the sense that no two power functions of a lesser algebraic degree can be used to represent the functions in this way.

We examine x^d to represent the composition of two functions which are different from those shown in [9] concerning binary weight. We establish that x^d is equivalent to a power function composed of a cubic power function and the inverse of a quadratic power function with six different representations in the following lemma. We see that the first two congruences are different from the four congruences given in [9].

Lemma 12. *The following equivalences hold for any positive integer* m*:*

$$\begin{aligned}
\textstyle\sum_{i=1}^{4} 2^{im} - 1 &\equiv 2^{3m+1}\tfrac{2^{4m}+2^m+1}{2^m+1} \\
&\equiv 2^{3m+1}\tfrac{2^{3m}+2^m+1}{2^{2m}+1} \\
&\equiv 2^{m+1}\tfrac{2^{2m}+2^m+1}{2^{4m}+1} \\
&\equiv 2^{m+1}\tfrac{2^{3m}+2^m+1}{2^{3m}+1} \\
&\equiv 2^{m+1}\tfrac{2^{3m}+2^{2m}+1}{2^{2m}+1} \\
&\equiv 2^{2m+1}\tfrac{2^{2m}+2^m+1}{2^m+1} \quad mod(2^{5m}-1)
\end{aligned}$$

Proof. Similar to [9], we consider first congruence and prove that $2^m + 1$ is invertible modulo $2^{5m} - 1$. To show that we need that $gcd(2^m + 1, 2^{5m} - 1)$. Then

$$gcd(2^i + 1, 2^j - 1) = \begin{cases} 1, & \text{if } j/gcd(i,j) \text{ is odd;} \\ 2^{gcd(i,j)} + 1, & \text{if } j/gcd(i,j) \text{ is even.} \end{cases}$$

We have $gcd(2^m + 1, 2^{5m} - 1) = 1$ since $gcd(m, 5m) = m$. If we replace 2^m by x, we have the equivalence $(x + 1)(x^4 + x^3 + x^2 + x - 1) \equiv 2x^3(x^4 + x + 1)$ mod $(x^5 - 1)$. Calculating the expression on the left-hand side of this equivalence, we have $(x + 1)(x^4 + x^3 + x^2 + x - 1) = x^5 + 2x^4 + 2x^3 + 2x^2 - 1 \equiv 2x^4 + 2x^3 + 2x^2$ mod $x^5 - 1$. Then considering the right-hand side we have $2x^7 + 2x^4 + 2x^3$ and we have $x^5 = 1$, so we have the equivalence.

We can prove other five statements of lemma similarly since $2^{2m} + 1$, $2^{3m} + 1$ and $2^{4m} + 1$ are invertible mod $(2^{5m} - 1)$. □

The corollary that follows is an obvious result of Lemma 12.

Corollary 7. Consider the power function x^d, where d is determined as $2^{4m} + 2^{3m} + 2^{2m} + 2^m - 1$,defined within the field $\mathbb{F}_{2^{5m}}$. It can be established that x^d exhibits cyclotomic equivalence to power functions characterized by the exponents $\frac{2^{4m}+2^m+1}{2^m+1}$, $\frac{2^{3m}+2^m+1}{2^{2m}+1}$, $\frac{2^{2m}+2^m+1}{2^{4m}+1}$, $\frac{2^{3m}+2^m+1}{2^{3m}+1}$, $\frac{2^{3m}+2^{2m}+1}{2^{2m}+1}$, $\frac{2^{2m}+2^m+1}{2^m+1}$. We know from [9] that this representations are optimal in terms of their weight.

We know that two power functions denoted as $F(x) = x^d$ and $G(x) = x^e$, defined over the finite field $\mathbb{F}_{2^n}$, where d, e, and n are positive integers, are considered cyclotomic equivalent under the following conditions:

- $d \equiv 2^k e \pmod{2^n - 1}$ for some positive integer k or
- $d^{-1} \equiv 2^k e \pmod{2^n - 1}$ for some positive integer k, but only if $gcd(d, 2^n - 1) = 1$, where d^{-1} represents the multiplicative inverse of d modulo $2^n - 1$.

Then we consider the multiplicative inverse d^{-1} of the power d and we try to represent d^{-1} as the fraction of two numbers by doing a computer search and we have the lemma as follows.

Lemma 13. *The following equivalents are true for any positive odd integer* m*:*

$$\begin{aligned}
\frac{1}{\sum_{i=1}^{4} 2^{im-1}} &\equiv 2^{3m-1}\frac{2^m+1}{2^{2m}+2^m+1} \\
&\equiv 2^{2m-1}\frac{2^{2m}+1}{2^{3m}+2^m+1} \\
&\equiv 2^{m-1}\frac{2^{3m}+1}{2^{3m}+2^{2m}+1} \\
&\equiv 2^{2m-1}\frac{2^m+1}{2^{4m}+2^m+1} \\
&\equiv 2^{m-1}\frac{2^{2m}+1}{2^{4m}+2^{2m}+1} \\
&\equiv 2^{m-1}\frac{2^m+1}{2^{4m}+2^{3m}+1} \quad \textit{mod } (2^{5m}-1)
\end{aligned}$$

Proof. We first prove that, $2^{2m} + 2^m + 1$, $2^{3m} + 2^m + 1$, $2^{3m} + 2^{2m} + 1$, $2^{4m} + 2^m + 1$, $2^{4m} + 2^{2m} + 1$ and $2^{4m} + 2^{3m} + 1$ are invertible modulo $2^{5m} - 1$.

We consider the first one and prove that

$$gcd(2^{2m} + 2^m + 1, 2^{5m} - 1) = 1$$

when m is odd. The others can be proven similarly. By Euclidean algorithm, we get the following equation array:

$$
\begin{aligned}
gcd(2^{2m}+2^m+1, 2^{5m}-1) &= gcd(2^{2m}+2^m+1, 2^{5m}-1 \bmod (2^{2m}+2^m+1)) \\
&= gcd(2^m+2, 2^{2m}-1) \\
&= gcd(2^m+2, 3).
\end{aligned}
$$

$$
gcd(2^m+2, 3) = \begin{cases} 1, & \text{if } m \text{ is odd;} \\ 3, & \text{if } m \text{ is even.} \end{cases}
$$

Hence, we see that they are relatively prime when m is odd. Then, the correctness holds as follows.

If we replace 2^m by x, we show that $(x+1)x^3(x^4+x^3+x^2+x-1) \equiv 2x^2+2x+2 \pmod{()x^5-1}$. For the left-hand side of this equation, we have $2x^7+2x^6+2x^5$ and this is equivalent to $2x^2+2x+2$ modulo (x^5-1).

The other congruences are proven in the same way. □

The following corollary follows from Lemma 13.

Corollary 8. Let x^d be the power function defined over the field $F_{2^{5m}}$ with $d = 2^{4m} + 2^{3m} + 2^{2m} + 2^m - 1$ and d^{-1} be the multiplicative inverse of d modulo $2^{5m}-1$. Then for any odd integer m, x^d is cyclotomic equivalent to the power functions with the exponents $\frac{2^m+1}{2^{2m}+2^m+1}$, $\frac{2^{2m}+1}{2^{3m}+2^m+1}$, $\frac{2^{3m}+1}{2^{3m}+2^{2m}+1}$, $\frac{2^m+1}{2^{4m}+2^m+1}$, $\frac{2^{2m}+1}{2^{4m}+2^{2m}+1}$ and $\frac{2^m+1}{2^{4m}+2^{3m}+1}$.

After mentioning all of the above on Dobbertin power function representations, we report our experimental results of the representations of this function in the next section.

4.2 Computational Results of Dobbertin Power Function's Representations

We have the Dobbertin function $F(x) = x^d$, $d = 2^{4m} + 2^{3m} + 2^{2m} + 2^m - 1$ where $n = 5m$. Then we try to find its cyclotomic equivalent function $G(x) = x^e$ satisfying the criteria $d \equiv 2^k e \pmod{()2^n-1}$ for some positive integer k, or $d^{-1} \equiv 2^k e \pmod{()2^n-1}$ for some positive integer k in the case that $gcd(d, 2^n-1) = 1$, with d^{-1} being the multiplicative inverse of d modulo 2^n-1.

- We can represent F(x) by combining a function with weight 4 and inverse of a

function with weight 3, for each k in $1 \leq k \leq n$. For $m = 2, 3, 4, 5, 6, 7$ we have 24 representations of the function.

- We can represent F(x) by combining a function with weight 5 and inverse of a function with weight 4, for each k in $1 \leq k \leq n$. For $m = 2$ we have 5 representations of the function, $m = 3$ and $m = 4$ we have 65 representations of the function.

- We can represent F(x) by combining a function with weight 4 and inverse of a function with weight 6, for each k in $1 \leq k \leq n$. For $m = 2$ we have 48 representations of the function, $m = 3$ we have 24 representations and $m = 4$ we have 44 representations of the function.

- We can represent F(x) by combining a function with weight 4 and inverse of a function with weight 7, for each k in $1 \leq k \leq n$. For $m = 2$ we have 12 representations of the function, $m = 3$ we have 184 representations and $m = 4$ we have 72 representations of the function.

- We can represent F(x) by combining a function with weight 5 and inverse of a function with weight 5, for each k in $1 \leq k \leq n$. For $m = 2$ we have 45 representations of the function, $m = 3$ we have 85 representations and $m = 4$ we have 55 representations of the function.

- We can represent F(x) by combining a function with weight 5 and inverse of a function with weight 7, for each k in $1 \leq k \leq n$. For $m = 2$ we have 5 representations of the function, $m = 3$ we have 220 representations of the function.

- We can represent F(x) by combining a function with weight 6 and inverse of a function with weight 4, for each k in $1 \leq k \leq n$. For $m = 2$ we have 72 representations of the function, $m = 3$ we have 204 representations and $m = 4$ we have 252 representations of the function.

- We can represent F(x) by combining a function with weight 7 and inverse of a function with weight 3, for each k in $1 \leq k \leq n$. For $m = 2$ we have 14 representations of the function, $m = 3$ we have 35 representations and $m = 4$ we have 21 representations of the function.

- We can represent F(x) by combining a function with weight 7 and inverse of a function with weight 5, for each k in $1 \leq k \leq n$. For $m = 2$ we have 14 representations of the function, $m = 3$ we have 462 representations of the function.

- We can represent F(x) by combining a function with weight 8 and inverse of a function with weight 4, for each k in $1 \leq k \leq n$. For $m = 2$ we have 8 representations of the function, $m = 3$ we have 72 representations of the function.

Then we try to find the representation of the function as a composition of two functions, one of which has negative terms. Our experimental result shows that there are no such representations.

Now, we present our experimental results of the representation x^d cyclotomic equivalent to the power functions with exponents e as a fraction of two integers x and y. Then we consider the cyclotomic equivalence $d \equiv 2^k e \pmod{()2^n - 1}$ for some positive integers d, e, n, k and examine $e = x/y$ with x has weight in range $(1, 12)$, and y has weight in range $(1, 12)$. In Figure 4.1, the location of the representations is marked in yellow for $m = 2$, $k = 1, 2, 3$.

After having these computational results, we consider representations of Niho power functions. In the next section, we review some computational results of Niho power function representations.

4.3 Computational Results of Niho Power Functions Representations

We consider Niho power function $F(x) = x^d$ with $d = 2^t + 2^{t/2} - 1$ defined over the field $\mathbb{F}_{2^{2t+1}}$, where t is an even integer. In this part of the study, we give our computational search results as follows.

- We can represent F(x) by combining a function with weight 2 and inverse of a function with weight 3, for each k in $1 \leq k \leq n$. For $m = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14$ we have 2 representations of the function.

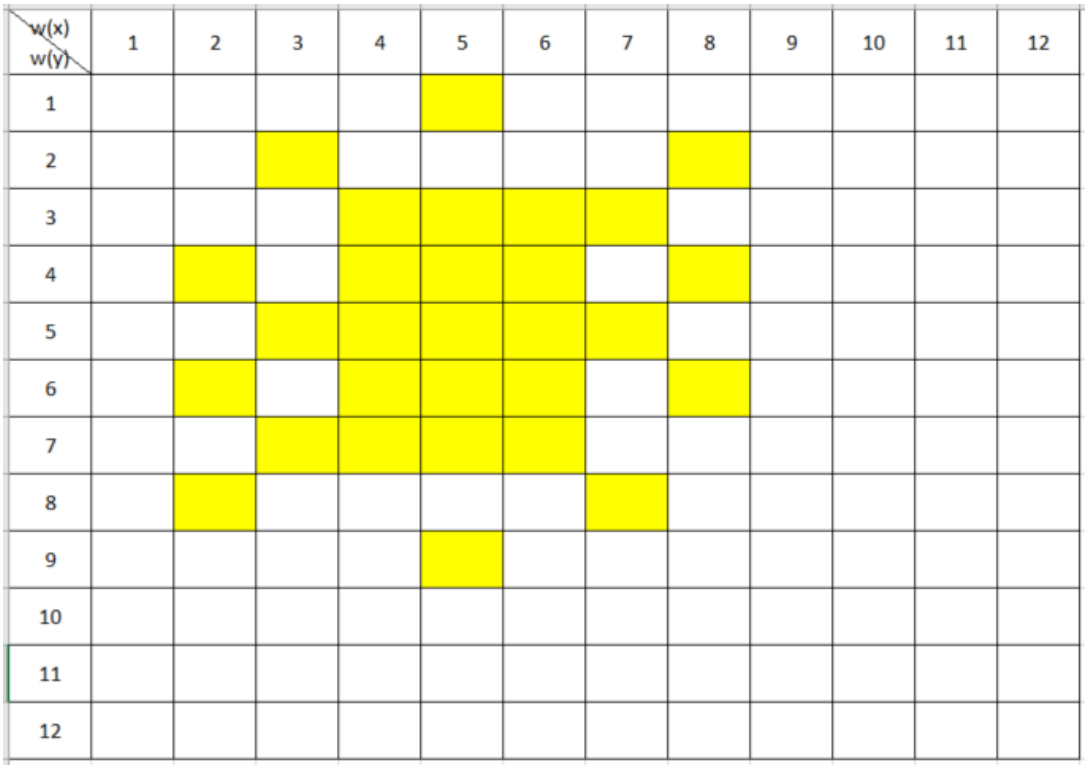

Figure 4.1: The location of the representations for $m = 2$, $k = 1, 2, 3$

- We can represent F(x) by combining a function with weight 2 and inverse of a function with weight 6, for each k in $1 \leq k \leq n$. For $m = 2, 3$ we have 4 representations and $m = 4, 5, 6$ we have 2 representations of the function.

- We can represent F(x) by combining a function with weight 3 and inverse of a function with weight 4, for each k in $1 \leq k \leq n$. For $m = 1, 2, 3, 4, 5, 6, 7$ we have 6 representations of the function.

- We can represent F(x) by combining a function with weight 3 and inverse of a function with weight 5, for each k in $1 \leq k \leq n$. For $m = 2, 3, 4, 5, 6$ we have 6 representations of the function.

- We can represent F(x) by combining a function with weight 3 and inverse of a function with weight 6, for each k in $1 \leq k \leq n$. For $m = 2$ we have 4 representations, $m = 3$ we have 9 and $m = 4, 5, 6$ we have 6 representations of the function.

- We can represent F(x) by combining a function with weight 3 and inverse of a function with weight 7, for each k in $1 \leq k \leq n$. For $m = 2$ we have 3

representations, $m = 3$ we have 12, $m = 4$ we have 15 representations and $m = 5$ we have 9 representations of the function.

- We can represent F(x) by combining a function with weight 4 and inverse of a function with weight 7, for each k in $1 \leq k \leq n$. For $m = 2$ we have 4 representations, $m = 3$ we have 36, $m = 4$ we have 60 representations of the function.

- We can represent F(x) by combining a function with weight 4 and inverse of a function with weight 6, for each k in $1 \leq k \leq n$. For $m = 2$ we have 4 representations, $m = 3$ we have 52, $m = 4$ we have 56 representations of the function.

- We can represent F(x) by combining a function with weight 4 and inverse of a function with weight 5, for each k in $1 \leq k \leq n$. For $m = 2$ we have 24 representations, $m = 3$ we have 28, $m = 4, 5$ we have 20 representations of the function.

- We can represent F(x) by combining a function with weight 4 and inverse of a function with weight 4, for each k in $1 \leq k \leq n$. For $m = 2$ we have 8 representations, $m = 3$ we have 16, $m = 4, 5$ we have 4 representations of the function.

- We can represent F(x) by combining a function with weight 5 and inverse of a function with weight 7, for each k in $1 \leq k \leq n$. For $m = 2$ we have 10 representations, $m = 3$ we have 115, $m = 4$ we have 290 representations of the function.

- We can represent F(x) by combining a function with weight 5 and inverse of a function with weight 6, for each k in $1 \leq k \leq n$. For $m = 2$ we have 10 representations, $m = 3$ we have 105, $m = 4$ we have 160 representations of the function.

- We can represent F(x) by combining a function with weight 5 and inverse of a function with weight 5, for each k in $1 \leq k \leq n$. For $m = 2$ we have 10 representations, $m = 3$ we have 75, $m = 4$ we have 60 representations of the function.

- We can represent F(x) by combining a function with weight 5 and inverse of a function with weight 4, for each k in $1 \leq k \leq n$. For $m = 2$ we have 30 representations, $m = 3$ we have 35, $m = 4$ we have 15 representations of the function.

- We can represent F(x) by combining a function with weight 6 and inverse of a function with weight 6, for each k in $1 \leq k \leq n$. For $m = 2$ we have 12 representations, $m = 3$ we have 144, $m = 4$ we have 384 representations of the function.

- We can represent F(x) by combining a function with weight 6 and inverse of a function with weight 5, for each k in $1 \leq k \leq n$. For $m = 2$ we have 12 representations, $m = 3$ we have 120, $m = 4$ we have 204 representations of the function.

- We can represent F(x) by combining a function with weight 6 and inverse of a function with weight 4, for each k in $1 \leq k \leq n$. For $m = 2$ we have 12 representations, $m = 3$ we have 102, $m = 4$ we have 90 representations and $m = 5$ we have 24 representations of the function.

- We can represent F(x) by combining a function with weight 6 and inverse of a function with weight 3, for each k in $1 \leq k \leq n$. For $m = 2$ we have 8 representations, $m = 3$ we have 18, $m = 4$ we have 60 representations and $m = 5$ we have 24 representations of the function.

- We can represent F(x) by combining a function with weight 6 and inverse of a function with weight 2, for each k in $1 \leq k \leq n$. For $m = 2, 3$ we have 6 representations, $m = 4$ we have 24 representations of the function.

- We can represent F(x) by combining a function with weight 7 and inverse of a function with weight 7, for each k in $1 \leq k \leq n$. For $m = 3$ we have 168 representations, $m = 4$ we have 1176 representations of the function.

- We can represent F(x) by combining a function with weight 7 and inverse of a function with weight 6, for each k in $1 \leq k \leq n$. For $m = 2$ we have 7 representations, $m = 3$ we have 203 representations and $m = 4$ we have 714 representations of the function.

- We can represent F(x) by combining a function with weight 7 and inverse of a function with weight 5, for each k in $1 \leq k \leq n$. For $m = 2$ we have 7 representations, $m = 3$ we have 182 representations and $m = 4$ we have 406 representations of the function.

- We can represent F(x) by combining a function with weight 7 and inverse of a function with weight 4, for each k in $1 \leq k \leq n$. For $m = 3$ we have 56 representations, $m = 4$ we have 280 representations of the function.

- We can represent F(x) by combining a function with weight 7 and inverse of a function with weight 3, for each k in $1 \leq k \leq n$. For $m = 2$ we have 14 representations, $m = 3$ we have 21 and $m = 4$ we have 42 representations of the function.

- We can represent F(x) by combining a function with weight 7 and inverse of a function with weight 2, for each k in $1 \leq k \leq n$. For $m = 3, 4$ we have 7 representations, $m = 5$ we have 35 representations of the function.

CHAPTER 5

LINEAR CODES GENERATED FROM WEAKLY REGULAR BENT FUNCTIONS OVER $\mathbb{F}_p$

5.1 The construction methods of linear codes from functions

Cryptographic functions have been a valuable resource in coding theory for creating linear codes with few weights for an extended period. The process of constructing linear codes using various types of functions such as quadratic, almost bent, (almost) perfect nonlinear, (weakly regular) bent, and plateaued functions has garnered significant attention in research.

One notable aspect is that determining the parameters of codes generated from these functions is comparatively straightforward, thanks to the well-structured nature of these functions. This contrasts with the typical challenges encountered in coding theory when dealing with parameter determination. Numerous techniques exist for constructing linear codes over finite fields. Among these methods, two stand out as generic, offering the potential to generate various classes of known codes. Let's explore these two generic construction methods for linear codes derived from functions:

First Generic Construction Method: For a polynomial $F(x)$ defined over the finite field $\mathbb{F}_q$, the first generic construction method for linear codes is defined as follows. Consider the linear codes $\mathcal{C}(F)$, which consist of codewords formed by applying the trace operator Tr^m to linear combinations of $F(x)$ and x for various choices of coefficients a and b in the field $\mathbb{F}_q$. Mathematically, this can be represented as:

$$\mathcal{C}(F) = \{(\mathrm{Tr}^m(aF(x) + bx))_{x \in \mathbb{F}_q^\star} : \ a, b \in \mathbb{F}_q\},$$

These linear codes have a length of $(q-1)$ and a dimension that is at most $2n$.

Second Generic Construction Method: The second generic construction method is based on a subset D defined as

$$\mathcal{C}_D = \{(\mathrm{Tr}^m(ad_1), \ldots, \mathrm{Tr}^m(ad_m)) \colon a \in \mathbb{F}_q\}. \tag{5.1.1}$$

In this case, the length of the codes is m, and the dimension is at most n. The quality of $\mathcal{C}_D$ depends on the choice of the defining set D.

The construction method represented by (5.1.1) was initially explored by Ding et al. [29, 30], resulting in the proposal of numerous linear codes, as documented in [27, 28, 29, 30, 31]. Furthermore, this construction method has been used to derive novel linear codes from cryptographic functions, as demonstrated in works such as [56, 52, 58, 66].

Inspired by the approach defined in (5.1.1), Li et al. [48] introduced a linear code construction based on a subset $\mathcal{D} = (x_1, y_1), \ldots, (x_m, y_m) \subseteq \mathbb{F}_q^2$. This linear code is defined as:

$$\mathcal{C}_\mathcal{D} = \mathbf{c}_{(a,b)} = (\mathrm{Tr}^n(ax_1 + by_1), \ldots, \mathrm{Tr}^n(ax_m + by_m)) \colon, a, b \in \mathbb{F}_q, \tag{5.1.2}$$

where the length of the code is m, and its dimension is at most $2n$. For every pair $(a, b) \in \mathbb{F}_q^2 \setminus (0, 0)$, the Hamming weight of a nonzero codeword $\mathbf{c}_{(a,b)}$ in $\mathcal{C}_\mathcal{D}$ is calculated as:

$$wt(\mathbf{c}_{(a,b)}) = \#\mathcal{D} - \mathcal{N}_\mathcal{D}(a, b),$$

where $\mathcal{N}_\mathcal{D}(a, b)$ represents the count of pairs $(x, y) \in \mathcal{D}$ such that the expression $\mathrm{Tr}^m(ax + by) = 0$. This construction method offers versatility in producing linear codes with different parameters, depending on the specific selection of the defining set $\mathcal{D}$.

Li et al. [48] successfully created several linear codes by utilizing the set $\mathcal{D}$, defined as $\mathcal{D} = \{(x, y) \in \mathbb{F}_q^2 \setminus (0, 0) \colon, \mathrm{Tr}^m(x^k + y^l) = 0\}$, where the parameters k and l belong to the sets $\{1, 2, p^{n/2} + 1\}$. In a more recent development, Jian et al. [42] extended these constructions by generating additional linear codes following the structure of

(5.1.2). They achieved this by employing a defining set $\mathcal{D}$ defined as $\mathcal{D} = \{(x, y) \in \mathbb{F}_q^2 \setminus (0, 0) \colon \mathrm{Tr}^m(x^k + y^{p^u+1}) = 0\}$, with k selected from the set $\{1, 2\}$. Subsequently, Wu et al. [61] introduced a novel set of linear codes following the structure presented in (5.1.2) and based on the set

$$\begin{aligned} \mathcal{D} &= \{(x, y) \in \mathbb{F}_q^2 \setminus \{(0, 0)\} \colon \mathrm{Tr}^m(x) + g(y) = 0\}, \\ \mathcal{D} &= \{(x, y) \in \mathbb{F}_q^2 \setminus \{(0, 0)\} \colon f(x) + g(y) = 0\}, \end{aligned} \tag{5.1.3}$$

here, the functions f and g represent weakly regular bent functions that map from $\mathbb{F}_q$ to $\mathbb{F}_p$. Moreover, Sınak [57] have recently constructed several linear codes of the form (5.1.2) based on the sets of the form (5.1.3) for weakly regular plateaued balanced and unbalanced functions. On the other hand, Cheng et al. [24] obtained several linear codes of the form (5.1.2) based on the following defining sets

$$\begin{aligned} \mathcal{D} &= \{(x, y) \in \mathbb{F}_q^2 \setminus \{(0, 0)\} \colon \mathrm{Tr}^m(x) + g(y) \in SQ\}, \\ \mathcal{D} &= \{(x, y) \in \mathbb{F}_q^2 \setminus \{(0, 0)\} \colon \mathrm{Tr}^m(x) + g(y) \in NSQ\}, \\ \mathcal{D} &= \{(x, y) \in \mathbb{F}_q^2 \setminus \{(0, 0)\} \colon f(x) + g(y) \in SQ\}, \\ \mathcal{D} &= \{(x, y) \in \mathbb{F}_q^2 \setminus \{(0, 0)\} \colon f(x) + g(y) \in NSQ\}, \end{aligned}$$

where f and g are weakly regular plateaued unbalanced functions. Motivated by the works [24, 42, 48, 57, 61]. In this thesis, we have embarked on the construction of linear codes in the style of Equation (5.1.2). These codes are constructed based on the defining sets outlined below:

$$\begin{aligned} \mathcal{D}_1 &= \{(x, y) \in \mathbb{F}_q^2 \colon \mathrm{Tr}^m(x) + g(y) = 1\}, \\ \mathcal{D}_2 &= \{(x, y) \in \mathbb{F}_q^2 \colon \mathrm{Tr}^m(x^2) + g(y) = 1\}, \\ \mathcal{D}_0 &= \{(x, y) \in (\mathbb{F}_q^2)^\star \colon \mathrm{Tr}^m(x^2) + g(y) = 0\}, \end{aligned} \tag{5.1.4}$$

where g is a weakly regular bent function from $\mathbb{F}_q$ to $\mathbb{F}_p$.

5.2 Three-weight linear codes based on the set $\mathcal{D}_1$

In this subsection, to construct linear codes over a finite field F_p, we consider the following defining set $\mathcal{D}_1 = \{(x, y) \in \mathbb{F}_q^2 \colon \mathrm{Tr}^m(x) + g(y) = 1\}$ when $g \in \mathcal{RF}$.

The following lemma finds the size of the defining set $\mathcal{D}_1$.

Lemma 14. *Let $\mathcal{D}_1$ be defined as in (5.1.4) and let $g \in \mathcal{RF}$. Then $\#\mathcal{D}_1 = p^{2m-1}$.*

Proof. From the orthogonality of exponential sums, we have

$$\begin{aligned}\#\mathcal{D}_1 &= \tfrac{1}{p}\sum_{x,y\in\mathbb{F}_q}\sum_{z\in\mathbb{F}_p}\xi_p^{z(\mathrm{Tr}^m(x)+g(y)-1)}\\ &= p^{2m-1}+\tfrac{1}{p}\sum_{z\in\mathbb{F}_p^\star}\sum_{x\in\mathbb{F}_q}\xi_p^{z\mathrm{Tr}^m(x)}\sum_{y\in\mathbb{F}_q}\xi_p^{zg(y)-z}\\ &= p^{2m-1}.\end{aligned}$$

□

We need the following lemma to find the Hamming weights of nonzero codewords in $\mathcal{C}_{\mathcal{D}_1}$.

Lemma 15. *Let $\mathcal{D}_1$ be defined as in (5.1.4). For $(a,b)\in(\mathbb{F}_q^2)^\star$, define $\mathcal{N}_{\mathcal{D}_1}(a,b) = \#\{(x,y)\in\mathcal{D}_1\colon \mathrm{Tr}^m(ax+by)=0\}$.*

- *If $a\in\mathbb{F}_q\setminus\mathbb{F}_p^\star$, we have that $\mathcal{N}_{\mathcal{D}_1}(a,b)=p^{2m-2}$.*
- *If $a\in\mathbb{F}_p^\star$, for even m*

$$\mathcal{N}_{\mathcal{D}_1}(a,b)=\begin{cases}p^{2m-2}+\epsilon_g(p-1)p^{m-2}\sqrt{p^*}^m, & \text{if } g^\star(a^{-1}b)=1,\\ p^{2m-2}-\epsilon_g p^{m-2}\sqrt{p^*}^m, & \text{if } g^\star(a^{-1}b)\neq 1,\end{cases}$$

and for odd m

$$\mathcal{N}_{\mathcal{D}_1}(a,b)=\begin{cases}p^{2m-2}, & \text{if } g^\star(a^{-1}b)=1,\\ p^{2m-2}+\epsilon_g p^{m-2}\sqrt{p^*}^{m+1}, & \text{if } g^\star(a^{-1}b)-1\in SQ,\\ p^{2m-2}-\epsilon_g p^{m-2}\sqrt{p^*}^{m+1}, & \text{if } g^\star(a^{-1}b)-1\in NSQ.\end{cases}$$

Proof. From the orthogonality of exponential sums, we have

$$\mathcal{N}_{\mathcal{D}_1}(a,b)=p^{2m-2}+\tfrac{1}{p^2}\sum_{z_1,z_2\in\mathbb{F}_p^\star}\xi_p^{-z_1}\sum_{x\in\mathbb{F}_q}\xi_p^{\mathrm{Tr}^m(z_1x+z_2ax)}\sum_{y\in\mathbb{F}_q}\xi_p^{z_1g(y)+\mathrm{Tr}^m(z_2by)}.$$

When $a\in\mathbb{F}_q\setminus\mathbb{F}_p^\star$, we can easily see that $\mathcal{N}_{\mathcal{D}_1}(a,b)=p^{2m-2}$. When $a\in\mathbb{F}_p^\star$, we have

the following proof. For $z_1 = -z_2 a$, we have

$$\begin{aligned}
\mathcal{N}_{\mathcal{D}_1}(a,b) &= p^{2m-2} + p^{m-2} \sum_{z_2 \in \mathbb{F}_p^\star} \xi_p^{z_2 a} \sum_{y \in \mathbb{F}_q} \xi_p^{-z_2 a g(y) + \mathrm{Tr}^m(z_2 b y)} \\
&= p^{2m-2} + p^{m-2} \sum_{z_2 \in \mathbb{F}_p^\star} \xi_p^{z_2 a} \sigma_{-z_2 a} \left(\sum_{y \in \mathbb{F}_q} \xi_p^{g(y) - \mathrm{Tr}^m(a^{-1} b y)} \right) \\
&= p^{2m-2} + p^{m-2} \sum_{z_2 \in \mathbb{F}_p^\star} \xi_p^{z_2 a} \sigma_{-z_2 a} \left(\epsilon_g \sqrt{p^*}^m \xi_p^{g^\star(a^{-1}b)} \right) \\
&= p^{2m-2} + p^{m-2} \epsilon_g \sqrt{p^*}^m \sum_{z_2 \in \mathbb{F}_p^\star} \eta_0^m(-z_2 a) \xi_p^{-z_2 a g^\star(a^{-1}b) + z_2 a}.
\end{aligned}$$

From Lemma 1, when m is even, we have

$$\begin{aligned}
\mathcal{N}_{\mathcal{D}_1}(a,b) &= p^{2m-2} + \epsilon_g p^{m-2} \sqrt{p^*}^m \sum_{z_2 \in \mathbb{F}_p^\star} \xi_p^{-z_2 a (g^\star(a^{-1}b) - 1)} \\
&= \begin{cases} p^{2m-2} + \epsilon_g (p-1) p^{m-2} \sqrt{p^*}^m, & \text{if } g^\star(a^{-1}b) = 1, \\ p^{2m-2} - \epsilon_g p^{m-2} \sqrt{p^*}^m, & \text{if } g^\star(a^{-1}b) \neq 1, \end{cases}
\end{aligned}$$

and when m is odd,

$$\begin{aligned}
\mathcal{N}_{\mathcal{D}_1}(a,b) &= p^{2m-2} + \epsilon_g p^{m-2} \sqrt{p^*}^m \sum_{z_2 \in \mathbb{F}_p^\star} \eta_0(-z_2 a) \xi_p^{-z_2 a (g^\star(a^{-1}b) - 1)} \\
&= \begin{cases} p^{2m-2}, & \text{if } g^\star(a^{-1}b) = 1, \\ p^{2m-2} + \epsilon_g p^{m-2} \sqrt{p^*}^{m+1}, & \text{if } g^\star(a^{-1}b) - 1 \in SQ, \\ p^{2m-2} - \epsilon_g p^{m-2} \sqrt{p^*}^{m+1}, & \text{if } g^\star(a^{-1}b) - 1 \in NSQ. \end{cases}
\end{aligned}$$

The proof is complete. □

We present the parameters of the code $\mathcal{C}_{\mathcal{D}_1}$ based on $\mathcal{D}_1$ in the following theorem.

Theorem 9. *Let $\mathcal{D}_1$ be defined as in (5.1.4) and $g \in \mathcal{RF}$. Then, the code $\mathcal{C}_{\mathcal{D}_1}$ of the form (5.1.2) is a 3-weight linear $[p^{2m-1}, 2m]$ code over $\mathbb{F}_p$. All parameters are listed in Tables 5.1 and 5.2 when m is even and odd, respectively.*

Table 5.1: The Hamming weights of $\mathcal{C}_{\mathcal{D}_1}$ in Theorem 9 when m is even

Hamming weight ω	Multiplicity A_ω
0	1
$(p-1)p^{2m-2}$	$p^{2m} - (p-1)p^m - 1$
$(p-1)(p^{2m-2} - \epsilon_g p^{m-2} \sqrt{p^*}^m)$	$\frac{(p-1)}{p}(p^m - \epsilon_g \sqrt{p^*}^m)$
$(p-1)p^{2m-2} + \epsilon_g p^{m-2} \sqrt{p^*}^m$	$\frac{(p-1)}{p}((p-1)p^m + \epsilon_g \sqrt{p^*}^m)$

Table 5.2: The Hamming weights of $\mathcal{C}_{\mathcal{D}_1}$ in Theorem 9 when m is odd

Hamming weight ω	Multiplicity A_ω
0	1
$(p-1)p^{2m-2}$	$p^{2m}-1+(p-1)(p^{m-1}-p^m+\epsilon\eta_0(-1)\sqrt{p^*}^{m-1})$
$(p-1)p^{2m-2}-\epsilon_g p^{m-2}\sqrt{p^*}^{m+1}$	$\frac{(p-1)}{2}(p^m-p^{m-1}-\epsilon\eta_0(-1)2\sqrt{p^*}^{m-1})$
$(p-1)p^{2m-2}+\epsilon_g p^{m-2}\sqrt{p^*}^{m+1}$	$\frac{(p-1)}{2}(p^m-p^{m-1})$

Proof. The proof follows from Lemmas 14 and 15. The length of the code $\mathcal{C}_{\mathcal{D}_1}$ is equal to the size of the defining set $\mathcal{D}_1$. From the definition, the Hamming weight of the zero codeword is zero, that is, $wt(\mathbf{c}_{(0,0)})=0$ when $(a,b)=(0,0)$. For every $(a,b)\in\mathbb{F}_q^2\setminus\{(0,0)\}$, the Hamming weight of the nonzero codeword $\mathbf{c}_{(a,b)}$ in $\mathcal{C}_{\mathcal{D}_1}$ is given by

$$wt(\mathbf{c}_{(a,b)})=\#\mathcal{D}_1-\mathcal{N}_{\mathcal{D}_1}(a,b).$$

When m is even, we have

$$wt(\mathbf{c}_{(a,b)}) = \begin{cases} (p-1)p^{2m-2}, & \text{if } a\in\mathbb{F}_q\setminus\mathbb{F}_p^\star,\\ (p-1)p^{2m-2}-\epsilon_g(p-1)p^{m-2}\sqrt{p^*}^m, & \text{if } a\in\mathbb{F}_p^\star \text{ and } g^\star(a^{-1}b)=1,\\ (p-1)p^{2m-2}+\epsilon_g p^{m-2}\sqrt{p^*}^m, & \text{if } a\in\mathbb{F}_p^\star \text{ and } g^\star(a^{-1}b)\neq 1, \end{cases}$$

and when m is odd, we have

$$wt(\mathbf{c}_{(a,b)}) = \begin{cases} (p-1)p^{2m-2}, & \text{if } a\in\mathbb{F}_p^\star \text{ and } g^\star(a^{-1}b)=1 \text{ or } a\in\mathbb{F}_q\setminus\mathbb{F}_p^\star,\\ (p-1)p^{2m-2}-\epsilon_g p^{m-2}\sqrt{p^*}^{m+1}, & \text{if } a\in\mathbb{F}_p^\star \text{ and } g^\star(a^{-1}b)-1\in SQ,\\ (p-1)p^{2m-2}+\epsilon_g p^{m-2}\sqrt{p^*}^{m+1}, & \text{if } a\in\mathbb{F}_p^\star \text{ and } g^\star(a^{-1}b)-1\in NSQ. \end{cases}$$

The weight distribution of this code follows from Lemma 7 with the help of the Pless power moments. Hence, we complete the proof. □

We provide the following example for the code $\mathcal{C}_{\mathcal{D}_1}$ given in Theorem 9, which is verified by MAGMA [3].

Example 1. *Let* $g:\mathbb{F}_{3^4}\to\mathbb{F}_3$ *be given as* $g(x)=\mathrm{Tr}^4(x^2)\in\mathcal{RF}$ *with* $\epsilon_g=-1$. *Then,* $\mathcal{C}_{\mathcal{D}_1}$ *is the* 3*-weight ternary linear* $[2187,8,1377]$ *code over* $\mathbb{F}_3$ *with the polynomial* $1+6398y^{1458}+60y^{1620}+102y^{1377}$.

5.3 Two-weight linear codes based on the set D_2

In this subsection, to construct linear codes over a finite field F_p, we consider the following defining set $\mathcal{D}_2=\{(x,y)\in\mathbb{F}_q^2:\ \mathrm{Tr}^m(x^2)+g(y)=1\}$ when $g\in\mathcal{RF}$.

The following lemma finds the size of the defining set $\mathcal{D}_2$.

Lemma 16. *Let $\mathcal{D}_2$ be defined as in (5.1.4) and let $g \in \mathcal{RF}$. Then*

$$\#\mathcal{D}_2 = \begin{cases} p^{2m-1} + \epsilon_g p^{m-1}, & \text{if } m \text{ is odd with } p \equiv 3 \pmod 4 \text{ or } m \text{ is even}, \\ p^{2m-1} - \epsilon_g p^{m-1}, & \text{if } m \text{ is odd with } p \equiv 1 \pmod 4. \end{cases}$$

Proof. The proof follows from Lemmas 1, 2, 3 and 4. Recall that we have $\mathcal{W}_g(0) = \epsilon_g \sqrt{p^*}^m$ by Lemma 4. From the orthogonality of exponential sums, we have

$$\begin{aligned} \#\mathcal{D}_2 &= \tfrac{1}{p} \sum_{x,y \in \mathbb{F}_q} \sum_{z \in \mathbb{F}_p} \xi_p^{z(\mathrm{Tr}^m(x^2)+g(y)-1)} \\ &= \tfrac{1}{p} \left(p^{2m} + \sum_{z \in \mathbb{F}_p^\star} \xi_p^{-z} \sum_{x \in \mathbb{F}_q} \xi_p^{\mathrm{Tr}^m(zx^2)} \sum_{y \in \mathbb{F}_q} \xi_p^{zg(y)} \right) \\ &= p^{2m-1} + \tfrac{1}{p} \sum_{z \in \mathbb{F}_p^\star} \xi_p^{-z} G(\eta)\eta(z)\sigma_z(\mathcal{W}_g(0)) \\ &= p^{2m-1} + \tfrac{1}{p} \sum_{z \in \mathbb{F}_p^\star} \xi_p^{-z} (-1)^{m-1} \sqrt{p^*}^m \eta_0^m(z) \epsilon_g \eta_0^m(z) \sqrt{p^*}^m \\ &= p^{2m-1} + \epsilon_g (-1)^m \eta_0^m(-1) p^{m-1}. \end{aligned}$$

Here, Lemmas 3 and 4 are used in the third equality, Lemma 2 is used in the fourth equality and Lemma 1 $(ii.)$ is used in the last equality. Thus the proof is complete. □

We need the following lemma to find the Hamming weights of nonzero codewords in $\mathcal{C}_{\mathcal{D}_2}$.

Lemma 17. *Let $\mathcal{D}_2$ be defined as in (5.1.4), and let $g \in \mathcal{RF}$ with $l_g = 2$. For $(a,b) \in (\mathbb{F}_q^2)^\star$, we define $\mathcal{N}_{\mathcal{D}_2}(a,b) = \#\{(x,y) \in \mathcal{D}_2 : \mathrm{Tr}^m(ax+by) = 0\}$. Then*

$$\mathcal{N}_{\mathcal{D}_2}(a,b) = \begin{cases} p^{2m-2} - \epsilon_g (-1)^{m-1} \eta_0^m(-1) p^{m-1}, & \text{if } t = 0 \text{ or } t \in NSQ, \\ p^{2m-2} + \epsilon_g (-1)^{m-1} \eta_0^m(-1) p^{m-1}, & \text{if } t \in SQ. \end{cases}$$

Proof. From the orthogonality of exponential sums, we get

$$\begin{aligned} \mathcal{N}_{\mathcal{D}_2}(a,b) &= \frac{1}{p^2} \sum_{x,y \in \mathbb{F}_q} \left(\sum_{z_1 \in \mathbb{F}_p} \xi_p^{z_1(\mathrm{Tr}^m(x^2)+g(y)-1)} \right) \left(\sum_{z_2 \in \mathbb{F}_p} \xi_p^{-z_2 \mathrm{Tr}^m(ax+by)} \right) \\ &= p^{2m-2} + \frac{1}{p^2}(A+B), \end{aligned}$$

where

$$\begin{aligned} A &= \sum_{z_1\in\mathbb{F}_p^\star}\sum_{x,y\in\mathbb{F}_q}\xi_p^{z_1(\mathrm{Tr}^m(x^2)+g(y)-1)} \\ B &= \sum_{z_1\in\mathbb{F}_p^\star}\xi_p^{-z_1}\sum_{z_2\in\mathbb{F}_p^\star}\sum_{x\in\mathbb{F}_q}\xi_p^{\mathrm{Tr}^m(z_1x^2-z_2ax)}\sum_{y\in\mathbb{F}_q}\xi_p^{z_1g(y)-\mathrm{Tr}^m(z_2by)}. \end{aligned}$$

From Lemma 16, one can observe that $A = p\#\mathcal{D}_2 - p^{2m}$, which implies that

$$A = \epsilon_g(-1)^m\eta_0^m(-1)p^m.$$

We can compute B with the help of algebraic tools given in Lemmas 1, 2, 3 and 4.

$$\begin{aligned} B &= \sum_{z_1\in\mathbb{F}_p^\star}\xi_p^{-z_1}\sum_{z_2\in\mathbb{F}_p^\star}\sum_{x\in\mathbb{F}_q}\xi_p^{\mathrm{Tr}^m(z_1x^2-z_2ax)}\sum_{y\in\mathbb{F}_q}\xi_p^{z_1g(y)-\mathrm{Tr}^m(z_2by)} \\ &= \sum_{z_1\in\mathbb{F}_p^\star}\xi_p^{-z_1}\sum_{z_2\in\mathbb{F}_p^\star}\sum_{x\in\mathbb{F}_q}\xi_p^{z_1(\mathrm{Tr}^m(x^2-\frac{z_2}{z_1}ax))}\sum_{y\in\mathbb{F}_q}\xi_p^{z_1(g(y)-\mathrm{Tr}^m(\frac{z_2}{z_1}by))} \\ &= \sum_{z_1\in\mathbb{F}_p^\star}\xi_p^{-z_1}\sum_{z_2\in\mathbb{F}_p^\star}\sum_{x\in\mathbb{F}_q}\xi_p^{z_1(\mathrm{Tr}^m(x^2-z_2ax))}\sigma_{z_1}\Big(\sum_{y\in\mathbb{F}_q}\xi_p^{g(y)-\mathrm{Tr}^m(z_2by)}\Big) \\ &= \sum_{z_1\in\mathbb{F}_p^\star}\xi_p^{-z_1}\sum_{z_2\in\mathbb{F}_p^\star}\sum_{x\in\mathbb{F}_q}\xi_p^{\mathrm{Tr}^m(z_1x^2-z_1z_2ax))}\sigma_{z_1}\big(\epsilon_g\sqrt{p^*}^m\xi_p^{g^\star(z_2b)}\big) \\ &= \sum_{z_1\in\mathbb{F}_p^\star}\xi_p^{-z_1}\sum_{z_2\in\mathbb{F}_p^\star}G(\eta)\eta(z_1)\xi_p^{-\mathrm{Tr}^m\left(\frac{(-z_1z_2a)^2}{4z_1}\right)}\epsilon_g\eta_0^m(z_1)\sqrt{p^*}^m\xi_p^{z_1g^\star(z_2b)}\big) \\ &= \epsilon_g(-1)^{m-1}p^{*m}\sum_{z_1\in\mathbb{F}_p^\star}\xi_p^{-z_1}\sum_{z_2\in\mathbb{F}_p^\star}\xi_p^{z_1g^\star(z_2b)-\mathrm{Tr}^m(\frac{z_1z_2^2a^2}{4})}\big) \\ &= \epsilon_g(-1)^{m-1}p^{*m}\sum_{z_1\in\mathbb{F}_p^\star}\xi_p^{-z_1}\sum_{z_2\in\mathbb{F}_p^\star}\xi_p^{z_1z_2^2(g^\star(b)-\mathrm{Tr}^m(\frac{a^2}{4}))}\big). \end{aligned}$$

Denote by $t = g^\star(b) - \mathrm{Tr}^m(\frac{a^2}{4})$. Remark that $t \in \mathbb{F}_p$ and it depends on the pairs $(a,b) \in (\mathbb{F}_q^2)^\star$.

- When $t = 0$, we have $B = -\epsilon_g(-1)^{m-1}(p-1)p^{*m}$.

- When $t \neq 0$, we have

$$\begin{aligned} B &= \epsilon_g(-1)^{m-1}p^{*m}\sum_{z_1\in\mathbb{F}_p^\star}\xi_p^{-z_1}\Big(\sum_{z_2\in\mathbb{F}_p}\xi_p^{z_1z_2^2(g^\star(b)-\mathrm{Tr}^m(\frac{a^2}{4}))}-1\Big)\\ &= \epsilon_g(-1)^{m-1}p^{*m}\sum_{z_1\in\mathbb{F}_p^\star}\xi_p^{-z_1}(\sqrt{p^*}\eta_0(z_1t)-1)\\ &= \epsilon_g(-1)^{m-1}p^{*m}\left(\sum_{z_1\in\mathbb{F}_p^\star}\eta_0(-z_1)\xi_p^{-z_1}\sqrt{p^*}\eta_0(-t)+1\right)\\ &= \epsilon_g(-1)^{m-1}p^{*m}\left(\sqrt{p^*}\sqrt{p^*}\eta_0(-t)+1\right)\\ &= \epsilon_g(-1)^{m-1}p^{*m}\left(p\eta_0(t)+1\right)\\ &= \begin{cases} \epsilon_g(-1)^{m-1}p^{*m}(p+1), & \text{if } \eta_0(t)=1,\\ -\epsilon_g(-1)^{m-1}p^{*m}(p-1), & \text{if } \eta_0(t)=-1. \end{cases} \end{aligned}$$

  In the second equality, we used the following fact

$$\textstyle\sum_{z_2\in\mathbb{F}_p}\xi_p^{z_1z_2^2t}=G(\eta_0)\eta_0(z_1t)=\sqrt{p^*}\eta_0(z_1t)$$

  from Lemmas 2 and 3, and in third equality we used Lemma 1.

Hence, we obtain that

$$\begin{aligned} \mathcal{N}_{\mathcal{D}_2}(a,b) &= p^{2m-2}+\tfrac{1}{p^2}(A+B)\\ &= \begin{cases} p^{2m-2}-\epsilon_g(-1)^{m-1}\eta_0^m(-1)p^{m-1}, & \text{if } t=0 \text{ or } t\in NSQ,\\ p^{2m-2}+\epsilon_g(-1)^{m-1}\eta_0^m(-1)p^{m-1}, & \text{if } t\in SQ, \end{cases} \end{aligned}$$

The proof is complete. □

We present the parameters of the code $\mathcal{C}_{\mathcal{D}_2}$ based on $\mathcal{D}_2$ in the following theorem.

Theorem 10. *Let $\mathcal{D}_2$ be defined as in (5.1.4) and $g \in \mathcal{RF}$ with $l_g = 2$. Then, the code $\mathcal{C}_{\mathcal{D}_2}$ of the form (5.1.2) is a 2-weight*

- *linear $[p^{2m-1}+\epsilon_g p^{m-1}, 2m]$ code with the parameters listed in Table 5.3 when m is odd with $p \equiv 3 \pmod 4$ or m is even,*
- *linear $[p^{2m-1}-\epsilon_g p^{m-1}, 2m]$ code with the parameters listed in Table 5.4 when m is odd with $p \equiv 1 \pmod 4$.*

Table 5.3: The Hamming weights of $\mathcal{C}_{\mathcal{D}_2}$ in Theorem 10 when m is odd with $p \equiv 3 \pmod 4$ or m is even

Hamming weight ω	Multiplicity A_ω
0	1
$(p-1)p^{2m-2}$	$p^{2m} - 1 - \frac{p-1}{2}p^{m-1}(p^m + \epsilon_g)$
$(p-1)p^{2m-2} + \epsilon_g 2p^{m-1}$	$\frac{p-1}{2}p^{m-1}(p^m + \epsilon_g)$

Table 5.4: The Hamming weights of $\mathcal{C}_{\mathcal{D}_2}$ in Theorem 10 when m is odd with $p \equiv 1 \pmod 4$

Hamming weight ω	Multiplicity A_ω
0	1
$(p-1)p^{2m-2}$	$p^{2m} - 1 - \frac{p-1}{2}p^{m-1}(p^m - \epsilon_g)$
$(p-1)p^{2m-2} - \epsilon_g 2p^{m-1}$	$\frac{p-1}{2}p^{m-1}(p^m - \epsilon_g)$

Proof. The proof follows from Lemmas 16 and 17. The length of the code $\mathcal{C}_{\mathcal{D}_2}$ is equal to the size of the defining set $\mathcal{D}_2$. The Hamming weight of the zero codeword is zero, that is, $wt(\mathbf{c}_{(0,0)}) = 0$ when $(a,b) = (0,0)$. For every $(a,b) \in \mathbb{F}_q^2 \setminus \{(0,0)\}$, the Hamming weight of the nonzero codeword $\mathbf{c}_{(a,b)}$ in $\mathcal{C}_{\mathcal{D}_2}$ is

$$wt(\mathbf{c}_{(a,b)}) \quad = \#\mathcal{D}_2 - \mathcal{N}_{\mathcal{D}_2}(a,b).$$

When m is odd with $p \equiv 3 \pmod 4$ or m is even,

$$wt(\mathbf{c}_{(a,b)}) \quad = \begin{cases} (p-1)p^{2m-2}, & \text{if } t = 0 \text{ or } t \in NSQ, \\ (p-1)p^{2m-2} + \epsilon_g 2p^{m-1}, & \text{if } t \in SQ. \end{cases}$$

When m is odd with $p \equiv 1 \pmod 4$,

$$wt(\mathbf{c}_{(a,b)}) \quad = \begin{cases} (p-1)p^{2m-2}, & \text{if } t = 0 \text{ or } t \in NSQ, \\ (p-1)p^{2m-2} - \epsilon_g 2p^{m-1}, & \text{if } t \in SQ. \end{cases}$$

The weight distribution of this code can be completely determined from Lemma 7 with the help of the Pless power moments, thereby completing the proof. □

We provide the following example for the code $\mathcal{C}_{\mathcal{D}_2}$ given in Theorem 10, which is verified by MAGMA [3].

Example 2. *Let $g : \mathbb{F}_{3^3} \to \mathbb{F}_3$ be given as $g(x) = \mathrm{Tr}^3(wx^2) \in \mathcal{RF}$, where $\epsilon_g = 1$ and w is a primitive element of $\mathbb{F}_{3^3}$. Then, $\mathcal{C}_{\mathcal{D}_2}$ is the 2-weight ternary linear $[234, 6, 144]$ code over $\mathbb{F}_3$ with the polynomial $1 + 494y^{162} + 234y^{144}$.*

5.4 Two-weight linear codes based on the set $\mathcal{D}_0$

In this subsection, to construct linear codes over a finite field F_p, we consider the following defining set $\mathcal{D}_0 = \{(x, y) \in (\mathbb{F}_q^2)^\star : \ \mathrm{Tr}^m(x^2) + g(y) = 0\}$ when $g \in \mathcal{RF}$. The following lemma finds the size of the defining set $\mathcal{D}_0$.

Lemma 18. *Let $\mathcal{D}_0$ be defined as in (5.1.4) and let $g \in \mathcal{RF}$. Then*

$$\#\mathcal{D}_0 = \begin{cases} p^{2m-1} - \epsilon_g(p-1)p^{m-1} - 1, & \text{if } m \text{ is even,} \\ p^{2m-1} + \epsilon_g\eta_0(-1)(p-1)p^{m-1} - 1, & \text{if } m \text{ is odd.} \end{cases}$$

Proof. The proof follows from Lemmas 1, 2, 3 and 4. From the orthogonality of exponential sums, we have

$$\begin{aligned} \#\mathcal{D}_0 + 1 &= \tfrac{1}{p} \sum_{x,y \in \mathbb{F}_q} \sum_{z \in \mathbb{F}_p} \xi_p^{z(\mathrm{Tr}^m(x^2)+g(y))} \\ &= \tfrac{1}{p} \left(p^{2m} + \sum_{z \in \mathbb{F}_p^\star} \sum_{x \in \mathbb{F}_q} \xi_p^{\mathrm{Tr}^m(zx^2)} \sum_{y \in \mathbb{F}_q} \xi_p^{zg(y)} \right) \\ &= p^{2m-1} + \tfrac{1}{p} \sum_{z \in \mathbb{F}_p^\star} G(\eta)\eta(z)\sigma_z(\mathcal{W}_g(0)) \\ &= p^{2m-1} + \tfrac{1}{p} \sum_{z \in \mathbb{F}_p^\star} (-1)^{m-1} \sqrt{p^*}^m \eta_0^m(z)\epsilon_g\eta_0^m(z)\sqrt{p^*}^m \\ &= p^{2m-1} + \epsilon_g(-1)^{m-1}\eta_0^m(-1)(p-1)p^{m-1}. \end{aligned}$$

Hence, the proof is complete. □

We need the following lemma to find the Hamming weights of nonzero codewords in $\mathcal{C}_{\mathcal{D}_0}$.

Lemma 19. *Let $\mathcal{D}_0$ be defined as in (5.1.4) and let $g \in \mathcal{RF}$ with $l_g = 2$. For $(a, b) \in (\mathbb{F}_q^2)^\star$, we define $\mathcal{N}_{\mathcal{D}_0}(a, b) = \#\{(x, y) \in \mathcal{D}_0 : \ \mathrm{Tr}^m(ax + by) = 0\}$. Then*

$$\mathcal{N}_{\mathcal{D}_0}(a, b) = \begin{cases} p^{2m-2} - 1 + \epsilon_g(-1)^{m-1}\eta_0^m(-1)(p-1)p^{m-1}, & \text{if } t = 0, \\ p^{2m-2} - 1, & \text{if } t \neq 0. \end{cases}$$

Proof. From the orthogonality of exponential sums, we get

$$\begin{aligned} \mathcal{N}_{\mathcal{D}_0}(a, b) + 1 &= \tfrac{1}{p^2} \sum_{x,y \in \mathbb{F}_q} \left(\sum_{z_1 \in \mathbb{F}_p} \xi_p^{z_1(\mathrm{Tr}^m(x^2)+g(y))} \right) \left(\sum_{z_2 \in \mathbb{F}_p} \xi_p^{-z_2\mathrm{Tr}^m(ax+by)} \right) \\ &= p^{2m-2} + \tfrac{1}{p^2}(A + B), \end{aligned}$$

where

$$\begin{aligned} A &= \sum_{z_1 \in \mathbb{F}_p^\star} \sum_{x,y \in \mathbb{F}_q} \xi_p^{z_1(\mathrm{Tr}^m(x^2)+g(y))}, \\ B &= \sum_{z_1,z_2 \in \mathbb{F}_p^\star} \sum_{x \in \mathbb{F}_q} \xi_p^{\mathrm{Tr}^m(z_1x^2 - z_2ax)} \sum_{y \in \mathbb{F}_q} \xi_p^{z_1 g(y) - \mathrm{Tr}^m(z_2by)}. \end{aligned}$$

By Lemma 18, we have $A = p(\#\mathcal{D}_0 + 1) - p^{2m}$, which implies that

$$A = \epsilon_g (-1)^{m-1} \eta_0^m(-1)(p-1)p^m.$$

We can compute B with the help of algebraic tools given in Lemmas 1, 2, 3 and 4.

$$\begin{aligned} B &= \sum_{z_1,z_2 \in \mathbb{F}_p^\star} \sum_{x \in \mathbb{F}_q} \xi_p^{\mathrm{Tr}^m(z_1x^2 - z_2ax)} \sum_{y \in \mathbb{F}_q} \xi_p^{z_1 g(y) - \mathrm{Tr}^m(z_2by)} \\ &= \sum_{z_1,z_2 \in \mathbb{F}_p^\star} \sum_{x \in \mathbb{F}_q} \xi_p^{z_1(\mathrm{Tr}^m(x^2 - \frac{z_2}{z_1}ax))} \sum_{y \in \mathbb{F}_q} \xi_p^{z_1(g(y) - \mathrm{Tr}^m(\frac{z_2}{z_1}by))} \\ &= \sum_{z_1,z_2 \in \mathbb{F}_p^\star} \sum_{x \in \mathbb{F}_q} \xi_p^{z_1(\mathrm{Tr}^m(x^2 - z_2ax))} \sigma_{z_1}\Big(\sum_{y \in \mathbb{F}_q} \xi_p^{g(y) - \mathrm{Tr}^m(z_2by)}\Big) \\ &= \sum_{z_1,z_2 \in \mathbb{F}_p^\star} \sum_{x \in \mathbb{F}_q} \xi_p^{\mathrm{Tr}^m(z_1x^2 - z_1z_2ax))} \sigma_{z_1}\big(\epsilon_g \sqrt{p^*}^m \xi_p^{g^\star(z_2b)}\big) \\ &= \sum_{z_1,z_2 \in \mathbb{F}_p^\star} G(\eta)\eta(z_1)\xi_p^{-\mathrm{Tr}^m\left(\frac{(-z_1z_2a)^2}{4z_1}\right)} \epsilon_g \eta_0^m(z_1)\sqrt{p^*}^m \xi_p^{z_1 g^\star(z_2b)}\big) \\ &= \epsilon_g(-1)^{m-1}p^{*m} \sum_{z_1,z_2 \in \mathbb{F}_p^\star} \xi_p^{z_1 g^\star(z_2b) - \mathrm{Tr}^m(\frac{z_1z_2^2a^2}{4})}\big) \\ &= \epsilon_g(-1)^{m-1}p^{*m} \sum_{z_1,z_2 \in \mathbb{F}_p^\star} \xi_p^{z_1z_2^2(g^\star(b) - \mathrm{Tr}^m(\frac{a^2}{4}))}\big). \end{aligned}$$

Denote by $t = g^\star(b) - \mathrm{Tr}^m(\frac{a^2}{4})$. Remark that $t \in \mathbb{F}_p$ and it depends on the pairs $(a,b) \in (\mathbb{F}_q^2)^\star$.

- When $t = 0$, $B = \epsilon_g(-1)^{m-1}\eta_0^m(-1)(p-1)^2p^m$.
- When $t \neq 0$,

$$\begin{aligned} B &= \epsilon_g(-1)^{m-1}p^{*m} \sum_{z_1 \in \mathbb{F}_p^\star} \Big(\sum_{z_2 \in \mathbb{F}_p} \xi_p^{z_1z_2^2(g^\star(b) - \mathrm{Tr}^m(\frac{a^2}{4}))} - 1\Big) \\ &= \epsilon_g(-1)^{m-1}p^{*m} \sum_{z_1 \in \mathbb{F}_p^\star} (\eta_0(z_1)\sqrt{p^*}\eta_0(t) - 1) \\ &= -\epsilon_g(-1)^{m-1}\eta_0^m(-1)(p-1)p^m, \end{aligned}$$

where we used the following fact $\sum_{z_2 \in \mathbb{F}_p} \xi_p^{z_1 t z_2^2} = G(\eta_0)\eta_0(z_1 t) = \sqrt{p^*}\eta_0(z_1 t)$ from Lemmas 3 and 2 in the second equality, and we used Lemma 1 in the third equality.

Hence, the proof is ended. □

We present the parameters of the code $\mathcal{C}_{\mathcal{D}_0}$ based on $\mathcal{D}_0$ in the following theorem.

Theorem 11. *Let $\mathcal{D}_0$ be defined as in (5.1.4) and $g \in \mathcal{RF}$ with $l_g = 2$. Then, $\mathcal{C}_{\mathcal{D}_0}$ is a 2-weight linear $[n, 2m]$ code over $\mathbb{F}_p$ with the parameters listed in Table 5.5, where*

$$n = \begin{cases} p^{2m-1} - \epsilon_g(p-1)p^{m-1} - 1, & \text{if } m \text{ is odd with } p \equiv 3 \pmod 4 \text{ or } m \text{ is even}, \\ p^{2m-1} + \epsilon_g(p-1)p^{m-1} - 1, & \text{if } m \text{ is odd with } p \equiv 1 \pmod 4. \end{cases}$$

Table 5.5: The Hamming weights of $\mathcal{C}_{\mathcal{D}_0}$ in Theorem 11

Hamming weight ω	Multiplicity A_ω
0	1
$(p-1)p^{2m-2}$	$p^{2m-1} - 1 + \epsilon_g(-1)^{m-1}\eta_0^m(-1)(p-1)p^{m-1}$
$(p-1)(p^{2m-2} + \epsilon_g(-1)^{m-1}\eta_0^m(-1)p^{m-1})$	$(p-1)(p^{2m-1} - \epsilon_g(-1)^{m-1}\eta_0^m(-1)p^{m-1})$

Proof. The proof follows from Lemmas 18 and 19. The length of the code $\mathcal{C}_{\mathcal{D}_0}$ is equal to the size of the defining set $\mathcal{D}_0$. From the definition, the Hamming weight of the zero codeword is zero, that is, $wt(\mathbf{c}_{(0,0)}) = 0$ when $(a, b) = (0, 0)$. For every $(a, b) \in \mathbb{F}_q^2 \setminus \{(0, 0)\}$, the Hamming weight of the nonzero codeword $\mathbf{c}_{(a,b)}$ in $\mathcal{C}_{\mathcal{D}_0}$ is

$$\begin{aligned} wt(\mathbf{c}_{(a,b)}) &= \#\mathcal{D}_0 - \mathcal{N}_{\mathcal{D}_0}(a, b) \\ &= \begin{cases} (p-1)p^{2m-2}, & \text{if } t = 0, \\ (p-1)p^{2m-2} + \epsilon_g(-1)^{m-1}\eta_0^m(-1)(p-1)p^{m-1}, & \text{if } t \neq 0. \end{cases} \end{aligned}$$

The weight distribution of the code $\mathcal{C}_{\mathcal{D}_0}$ can be completely determined from Lemma 7 with the help of the Pless power moments, which completes the proof. □

We provide the following example for the code $\mathcal{C}_{\mathcal{D}_0}$ given in Theorem 11, which is verified by MAGMA [3].

Example 3. *Let $g : \mathbb{F}_{3^2} \to \mathbb{F}_3$ be given as $g(x) = \mathrm{Tr}^2(x^{10}) \in \mathcal{RF}$ with $\epsilon_g = 1$. Then, $\mathcal{C}_{\mathcal{D}_0}$ is the 2-weight ternary linear $[20, 6, 12]$ code with the polynomial $1 + 20y^{18} + 60y^{12}$.*

CHAPTER 6

CONCLUSION

This chapter provides a summary of the study presented in this thesis and discusses the need for additional research in this area.

In this thesis, we divided our work into three parts. The first one is about necessary and sufficient conditions on some families of bivariate APN functions built by Carlet's bivariate APN construction. In the second part, we presented our conclusions about the representation of Dobbertin and Niho power functions and provided relevant experimental data. Then in the third part, we report our study on few-weight linear codes based on weakly regular bent functions over finite fields.

To give all the information above, prior to the subject, we started with the preliminary information. Since APN and bent functions are vectorial Boolean functions, we started by describing vectorial Boolean functions and reporting properties of them. We gave information about how vectorial Boolean functions are used in block ciphers, and then we mentioned about representations of these functions. Then differential uniformity and nonlinearity of these functions are provided and after that related equivalence relations are described. Then the details regarding APN Power functions and known infinite families of Non-power APN functions are given. And last part of the preliminary information is about linear codes based on weakly regular bent functions over finite fields.

For the first part, we described Carlet's bivariate APN construction in which APN functions are built by using the simplest Maiorana–McFarland function. Considering Theorem 4 we gave necessary and sufficient conditions on functions F_1, F_2 and F_3

to be APN. Then we dealt with four functions defined in Theorem 8 and we demonstrated the conditions in Carlets construction for these functions to be APN in special cases.

In the second part of this thesis, we considered representations of Dobbertin and Niho APN power functions. We reported our conclusions about this work then we did some experimentation to represent these APN power functions and gave our related experimental results on it.

And third part is about the construction of linear codes from cryptographic functions namely weakly regular bent functions. We presented several linear codes with flexible parameters by employing weakly regular bent functions.

We note certain limitations in Chapter 5. Addressing these limitations and exploring the possibility of generating new linear codes through alternative cryptographic functions within the constructions described earlier could be potential avenues for future research.

As a future work, we aim to construct a new code $\mathcal{C}_{\mathcal{D}_{fg}}$ based on the defining set $\mathcal{D}_{fg} = \{(x, y) \in \mathbb{F}_q^2 \colon\ f(x) + g(y) = 1\}$ for given two weakly regular bent functions $f, g \in \mathcal{RF}$.

REFERENCES

[1] T. Beth and C. Ding, On almost perfect nonlinear permutations, Proceedings of Workshop on the Theory and Application of of Cryptographic Techniques, pp. 65–76, 1994.

[2] E. Biham and A. Shamir, Differential cryptanalysis of DES-like cryptosystems, Journal of Cryptology, 4(1), pp. 3–72, 1991.

[3] W. Bosma, J. Cannon, and C. Playoust, The Magma algebra system. I. The user language, J. Symbolic Comput., 24(3-4), pp. 235–265, 1997, ISSN 0747-7171.

[4] C. Bracken, E. Byrne, N. Markin, and G. McGuire, New families of quadratic almost perfect nonlinear trinomials and multinomials, Finite Fields and Their Applications, 14(3), pp. 703–714, 2008.

[5] C. Bracken, E. Byrne, N. Markin, and G. McGuire, A few more quadratic APN functions, Cryptography and Communications, 3, pp. 43–53, 2011.

[6] K. Browning, APN polynomials and related codes, Special volume of Journal of Combinatorics, Information and System Sciences, 34, pp. 135–159, 2009.

[7] L. Budaghyan, *Construction and analysis of cryptographic functions*, Springer, 2015.

[8] L. Budaghyan, M. Calderini, C. Carlet, R. S. Coulter, and I. Villa, Constructing apn functions through isotopic shifts, IEEE Transactions on Information Theory, 66(8), pp. 5299–5309, 2020.

[9] L. Budaghyan, M. Calderini, C. Carlet, D. Davidova, and N. S. Kaleyski, On two fundamental problems on APN power functions, IEEE Transactions on Information Theory, 68(5), pp. 3389–3403, 2022.

[10] L. Budaghyan, M. Calderini, and I. Villa, On relations between CCZ-and EA-equivalences, Cryptography and Communications, 12(1), pp. 85–100, 2020.

[11] L. Budaghyan and C. Carlet, Classes of quadratic APN trinomials and hexanomials and related structures, IEEE Transactions on Information Theory, 54(5), pp. 2354–2357, 2008.

[12] L. Budaghyan, C. Carlet, and G. Leander, A class of quadratic APN binomials inequivalent to power functions, Cryptology ePrint Archive, 2006.

[13] L. Budaghyan, C. Carlet, and G. Leander, Two classes of quadratic APN binomials inequivalent to power functions, IEEE Transactions on Information Theory, 54(9), pp. 4218–4229, 2008.

[14] L. Budaghyan, C. Carlet, and G. Leander, Constructing new APN functions from known ones, Finite Fields and Their Applications, 15(2), pp. 150–159, 2009.

[15] L. Budaghyan, C. Carlet, and G. Leander, On a construction of quadratic APN functions, in *2009 IEEE Information Theory Workshop*, pp. 374–378, IEEE, 2009.

[16] L. Budaghyan, C. Carlet, and A. Pott, New classes of almost bent and almost perfect nonlinear polynomials, IEEE Transactions on Information Theory, 52(3), pp. 1141–1152, 2006.

[17] L. Budaghyan, T. Helleseth, and N. Kaleyski, A new family of APN quadrinomials, IEEE Transactions on Information Theory, 66(11), pp. 7081–7087, 2020.

[18] M. Calderini, L. Budaghyan, and C. Carlet, On known constructions of APN and AB functions and their relation to each other, Rad Hrvatske akademije znanosti i umjetnosti: Matematičke znanosti, (546=25), pp. 79–105, 2021.

[19] C. Carlet, Relating three nonlinearity parameters of vectorial functions and building APN functions from bent functions, Designs, Codes and Cryptography, 59(1), pp. 89–109, 2011.

[20] C. Carlet, *Boolean functions for cryptography and coding theory*, Cambridge University Press, 2021.

[21] C. Carlet, P. Charpin, and V. Zinoviev, Codes, bent functions and permutations suitable for DES-like cryptosystems, Designs, Codes and Cryptography, 15(2), pp. 125–156, 1998.

[22] C. Carlet, C. Ding, and J. Yuan, Linear codes from perfect nonlinear mappings and their secret sharing schemes, IEEE Transactions on Information Theory, 51(6), pp. 2089–2102, 2005.

[23] F. Chabaud and S. Vaudenay, Links between differential and linear cryptanalysis, pp. 356–365, 1995.

[24] Y. Cheng and X. Cao, Linear codes with few weights from weakly regular plateaued functions, Discrete Mathematics, 344(12), p. 112597, 2021.

[25] J. Daemen and V. Rijmen, The design of Rijndael: AES-the advanced encryption standard, vol, 23, pp. 362–366, 2013.

[26] D. Davidova, On properties of bent and almost perfect nonlinear functions, 2021.

[27] C. Ding, Linear codes from some 2-designs, IEEE Transactions on information theory, 61(6), pp. 3265–3275, 2015.

[28] C. Ding, A construction of binary linear codes from boolean functions, Discrete mathematics, 339(9), pp. 2288–2303, 2016.

[29] C. Ding and H. Niederreiter, Cyclotomic linear codes of order 3, IEEE Transactions on information theory, 53(6), pp. 2274–2277, 2007.

[30] C. Ding and J. Yuan, Covering and secret sharing with linear codes, in *International Conference on Discrete Mathematics and Theoretical Computer Science*, pp. 11–25, Springer, 2003.

[31] K. Ding and C. Ding, A class of two-weight and three-weight codes and their applications in secret sharing, IEEE Transactions on Information Theory, 61(11), pp. 5835–5842, 2015.

[32] H. Dobbertin, Almost perfect nonlinear power functions on GF(2^n): the Niho case, Information and Computation, 151(1-2), pp. 57–72, 1999.

[33] H. Dobbertin, Almost perfect nonlinear power functions on GF(2^n): the Welch case, IEEE Transactions on Information Theory, 45(4), pp. 1271–1275, 1999.

[34] H. Dobbertin, Almost perfect nonlinear power functions on GF(2^n): A new case for n divisible by 5, Finite Fields and Applications, pp. 113–121, 2001.

[35] H. Dobbertin, Almost perfect nonlinear power functions on GF(2^n): a new case for n divisible by 5, Finite Fields and Applications: Proceedings of The Fifth International Conference on Finite Fields and Applications F q 5, held at the University of Augsburg, Germany, August 2–6, 1999, pp. 113–121, 2001.

[36] Y. Edel, G. Kyureghyan, and A. Pott, A new APN function which is not equivalent to a power mapping, IEEE Transactions on Information Theory, 52(2), pp. 744–747, 2006.

[37] R. Gold, Maximal recursive sequences with 3-valued recursive cross-correlation functions (corresp.), IEEE transactions on Information Theory, 14(1), pp. 154–156, 1968.

[38] F. Göloğlu, Gold-hybrid APN functions, Preprint, 2020.

[39] Z. Heng, D. Li, and F. Liu, Ternary self-orthogonal codes from weakly regular bent functions and their application in lcd codes, Designs, Codes and Cryptography, pp. 1–24, 2023.

[40] W. C. Huffman and V. Pless, *Fundamentals of error-correcting codes*, Cambridge university press, 2010.

[41] H. Janwal and R. Wilsonat, Hyperplane sections of fermat varieties in P3 in char. 2 and some applications to cyclic, 673, p. 180, 1993.

[42] G. Jian, Z. Lin, and R. Feng, Two-weight and three-weight linear codes based on weil sums, Finite Fields and Their Applications, 57, pp. 92–107, 2019.

[43] T. Kasami, The weight enumerators for several classes of subcodes of the 2nd order binary Reed-Muller codes, Information and Control, 18(4), pp. 369–394, 1971.

[44] L. R. Knudsen, Truncated and higher order differentials, Fast Software Encryption: Second International Workshop Leuven, Belgium, December 14–16, 1994 Proceedings 2, pp. 196–211, 1995.

[45] P. V. Kumar, R. A. Scholtz, and L. R. Welch, Generalized bent functions and their properties, Journal of Combinatorial Theory, Series A, 40(1), pp. 90–107, 1985.

[46] G. Lachaud and J. Wolfmann, The weights of the orthogonals of the extended quadratic binary Goppa codes, IEEE transactions on information theory, 36(3), pp. 686–692, 1990.

[47] C. Li, N. Li, T. Helleseth, and C. Ding, The weight distributions of several classes of cyclic codes from apn monomials, IEEE transactions on information theory, 60(8), pp. 4710–4721, 2014.

[48] C. Li, Q. Yue, and F.-W. Fu, A construction of several classes of two-weight and three-weight linear codes, Applicable Algebra in Engineering, Communication and Computing, 28, pp. 11–30, 2017.

[49] R. Lidl and H. Niederreiter, *Finite fields*, 20, Cambridge university press, 1997.

[50] S. Mesnager, Linear codes with few weights from weakly regular bent functions based on a generic construction, Cryptography and Communications, 9, pp. 71–84, 2017.

[51] S. Mesnager, F. Özbudak, and A. Sınak, Linear codes from weakly regular plateaued functions and their secret sharing schemes, Designs, Codes and Cryptography, 87(2-3), pp. 463–480, 2019.

[52] S. Mesnager and A. Sınak, Several classes of minimal linear codes with few weights from weakly regular plateaued functions, IEEE Transactions on Information Theory, 66(4), pp. 2296–2310, 2019.

[53] K. Nyberg, Differentially uniform mappings for cryptography, Workshop on the Theory and Application of of Cryptographic Techniques, pp. 55–64, 1994.

[54] O. S. Rothaus, On "bent" functions, Journal of Combinatorial Theory, Series A, 20(3), pp. 300–305, 1976.

[55] A. Sınak, Minimal linear codes from weakly regular plateaued balanced functions, Discrete Mathematics, 344(3), p. 112215, 2021.

[56] A. SINAK, Minimal linear codes with six-weights based on weakly regular plateaued balanced functions, International Journal of Information Security Science, 10(3), pp. 86–98, 2021.

[57] A. Sınak, Construction of minimal linear codes with few weights from weakly regular plateaued functions., Turkish Journal of Mathematics, 46(3), pp. 953–972, 2022.

[58] C. Tang, N. Li, Y. Qi, Z. Zhou, and T. Helleseth, Linear codes with two or three weights from weakly regular bent functions, IEEE Transactions on Information Theory, 62(3), pp. 1166–1176, 2016.

[59] H. Taniguchi, On some quadratic APN functions, Designs, Codes and Cryptography, 87(9), pp. 1973–1983, 2019.

[60] The Sage Developers, *SageMath, the Sage Mathematics Software System (Version 9.5)*, 2023, `https://www.sagemath.org`.

[61] Y. Wu, N. Li, and X. Zeng, Linear codes with few weights from cyclotomic classes and weakly regular bent functions, Designs, Codes and Cryptography, 88, pp. 1255–1272, 2020.

[62] S. Yoshiara, Equivalences of quadratic APN functions, Journal of Algebraic Combinatorics, 35(3), pp. 461–475, 2012.

[63] S. Yoshiara, Equivalences of power APN functions with power or quadratic APN functions, Journal of Algebraic Combinatorics, 44, pp. 561–585, 2016.

[64] X. Zeng, J. Shan, and L. Hu, A triple-error-correcting cyclic code from the gold and kasami–welch apn power functions, Finite Fields and Their Applications, 18(1), pp. 70–92, 2012.

[65] Y. Zhou and A. Pott, A new family of semifields with 2 parameters, Advances in Mathematics, 234, pp. 43–60, 2013.

[66] Z. Zhou, N. Li, C. Fan, and T. Helleseth, Linear codes with two or three weights from quadratic bent functions, Designs, Codes and Cryptography, 81, pp. 283–295, 2016.

APPENDIX A

SAGEMATH IMPLEMENTATIONS OF ALGORITHMS IN THIS STUDY

In this appendix, we present our SageMath [60] script that we implement to find representations of Dobbertin and Niho functions as the composition of two functions like $x^i \circ x^{1/j}$.

```
def findPattern(m):
    n = 5*m
    d = 2^(4*m)+2^(3*m)+2^(2*m)+2^(m)-1
    b=0
    print("d=",d)
    for k in range(1,n):
        print ("k:",k,)
        c=0
        for j in range(0,n):
            for i in range(0,j):
                for l in range(1,n):
                    for r in range(1,l):
                        for s in range(1):
                            p1=2^l
                            p2=2^r
                            p3=2^s
                            pay=p1 + p2 + p3
                            val =(d*(2^j+2^i)-2^k*pay) % (2^(n)-1)
                            if val == 0:
                                print("The values are: j=", j,", i=", i, ",l=", l, ",r=", r, ",s=", s)
                                print("RESULT:", pay/(2^j+2^i))
                                binary = format(pay,'b')
                                print("weight: ", binary)
                                c=c+1
                                b=b+1
        print("Number of kResults: ",c,)
    print("Number of Results: ",b,)
for m in range(2,18):
    print("m:",m,)
    findPattern(m)
```

Figure A.1: Representation of Dobbertin Function by combining a function with weight 3 and inverse of a function with weight 2

```
def findPattern(m):
    n = 5*m
    d = 2^(4*m)+2^(3*m)+2^(2*m)+2^(m)-1
    b=0
    print("d=",d)
    for k in range(1,n):
        print ("k:",k,)
        c=0
        for j in range(0,n):
            for i in range(0,j):
                for f in range(0,i):
                    for l in range(1,n):
                        for r in range(1,l):
                            for s in range(1,r):
                                for t in range(1):
                                    p1=2^l
                                    p2=2^r
                                    p3=2^s
                                    p4=2^t
                                    pay=p1 + p2 + p3 + p4
                                    val =(d*(2^j+2^i+2^f)-2^k*pay) % (2^(n)-1)
                                    if val == 0:
                                        print("The values are: j=", j,", i=", i,", f=", f, ",l=", l, ",r=", r, ",s=", s, ",t=", t
                                        print("RESULT:", pay/(2^j+2^i+2^f))
                                        binary = format(pay,'b')
                                        print("weight: ", binary)
                                        c=c+1
                                        b=b+1
        print("Number of kResults: ",c,)
    print("Number of Results: ",b,)
for m in range(2,18):
    print("m:",m,)
    findPattern(m)
```

Figure A.2: Representation of Dobbertin Function by combining a function with weight 4 and inverse of a function with weight 3

```
def findPattern(m):
    n = 4*m+1
    d = 2^(2*m)+2^(m)-1
    b=0
    print("d=",d)
    for k in range(1,n):
        print("k: ",k,)
        c=0
        for j in range(0,n):
            for i in range(0,j):
                for f in range(0,i):
                    for l in range(1,n):
                        for r in range(1):
                            p1=2^l
                            p2=2^r
                            pay=p1+p2
                            val =(d*(2^j+2^i+2^f)-2^k*pay) % (2^(n)-1)
                            if val == 0:
                                print("The values are: j=", j,", i=", i,", f=", f, ", l=", l, ", r=", r)
                                print("RESULT:", pay/(2^j+2^i+2^f))
                                binary = format(pay,'b')
                                print("weight: ", binary)
                                c=c+1
                                b=b+1
        print("Number of kResults: ",c,)
    print("Number of Results: ",b,)
for m in range(2,18):
    print("m:",m,)
    t=2*m
    print("t:",t,)
    findPattern(m)
```

Figure A.3: Representation of Niho Function by combining a function with weight 2 and inverse of a function with weight 3

```
def findPattern(m):
    n = 4*m+1
    d = 2^(2*m)+2^(m)-1
    b=0
    print("d=",d)
    for k in range(1,n):
        print("k: ",k,)
        c=0
        for j in range(0,n):
            for i in range(0,j):
                for f in range(0,i):
                    for g in range(0,f):
                        for l in range(1,n):
                            for r in range(1,l):
                                for s in range(1):
                                    p1=2^l
                                    p2=2^r
                                    p3=2^s
                                    pay=p1+p2+p3
                                    val =(d*(2^j+2^i+2^f+2^g)-2^k*pay) % (2^(n)-1)
                                    if val == 0:
                                        print("The values are: j=", j,", i=", i, ", l=", l, ", r=", r, ", s=", s, ", f=", f, ",
                                        print("RESULT:", pay/(2^j+2^i+2^f+2^g))
                                        binary = format(pay,'b')
                                        print("weight: ", binary)
                                        c=c+1
                                        b=b+1
        print("Number of kResults: ",c,)
    print("Number of Results: ",b,)
for m in range(1,18):
    print("m:",m,)
    t=2*m
    print("t:",t,)
    findPattern(m)
```

Figure A.4: Representation of Niho Function by combining a function with weight 3 and inverse of a function with weight 4

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Acunalp Erleblebici, İlksen
Nationality: Turkish

EDUCATION

Degree	Institution	Year of Graduation
M.S.	Institute of Applied Mathematics, METU	2007
B.S.	Department of Mathematics, METU	2005
High School	Karatay Lisesi	2000

PROFESSIONAL EXPERIENCE

Year	Place	Enrollment
2007-2019	TÜBİTAK	Chief Researcher
2019-2021	HAVELSAN	Project Monitoring Engineer
2023-	TÜBİTAK	Consultant Researcher

PUBLICATIONS

International Conference Publications

İ. Acunalp Erleblebici, O. Yayla, *A General Version of Carlet's Construction of APN Functions*, 27th International Conference on Applications of Computer Algebra, 2022, Gebze, Türkiye.

İ. Acunalp Erleblebici, A. Sınak, O. Yayla, *On the Carlet's Bivariate APN Construction and Representations of Dobbertin Power Function*, preprint

İ. Acunalp Erleblebici, A. Sınak, O. Yayla, *Few-weight linear codes based on weakly regular bent functions over finite fields*, preprint