MASTER THESIS

Jaroslav Kroutil

Computational methods for finding cryptographic functions

Department of Algebra

Supervisor of the master thesis:	Dr. rer. nat. Faruk Göloglu
Study programme:	Mathematics
Study branch:	Mathematics for Information Technologies

Prague 2024

Title: Computational methods for finding cryptographic functions

Author: Jaroslav Kroutil

Department: Department of Algebra

Supervisor: Dr. rer. nat. Faruk Göloglu, Department of Algebra

Abstract: Recent studies have demonstrated several methods on different approaches to classification of vectorial Boolean functions up to certain equivalence relation and to finding new quadratic Almost Perfect Nonlinear (APN) functions. In this work we explore these classification methods of vectorial Boolean functions, in particular those that minimise the search space up to EA-equivalence or linear-equivalence. We also investigate various strategies for finding quadratic APN functions. These methods are rooted in various aspects of algebraic theory. We explore the mathematical theory in more detail, and provide a guide to practical application of the theory. We also provide implementations of these methods and illustrate them in the context of the presented theory.

Keywords: Boolean functions, APN, equivalence, quadratic, computational methods

Contents

Introduction

S-boxes, or substitution boxes, play a crucial role in modern symmetric cryptographic systems. They are primarily used to introduce non-linearity into the encryption process, which is essential to resist linear and differential cryptanalysis attacks.

The relationship between S-boxes and vectorial Boolean functions is quite profound. An S-box can be thought of as a vectorial Boolean function, which is a function that maps an n-dimensional Boolean vector to an m-dimensional Boolean vector. The properties of these functions, such as non-linearity, algebraic degree, and differential uniformity, combined affect the strength of the S-box and, consequently, the security of the entire cryptographic system.

Almost Perfect Nonlinear (APN) functions, are a special class of vectorial Boolean functions. They have the best possible differential uniformity, a property that is highly desirable for S-boxes. This makes it extremely difficult for an attacker to derive the key based on the differences between pairs of plaintext and ciphertext. However, designing S-boxes based on APN functions is a complex task and is a the subject of ongoing research in cryptography.

Equivalences between vectorial Boolean functions are of a great importance in the study of S-boxes. Two vectorial Boolean functions are equivalent if one can be obtained from the other by a combination of affine transformations. Understanding the structure of these equivalences can lead to the discovery of new, more efficient, and more secure cryptographic primitives and it also helps to simplify the design and analysis of S-boxes.

The advent of quantum computers poses a new set of challenges for symmetric cryptography. Quantum computers can potentially perform attacks on cryptographic systems more efficiently than classical computers. For example, Grover's algorithm [1], can significantly speed up the brute-force search for a cryptographic key. However, it is worth noting that symmetric cryptographic systems like those using S-boxes are generally more resistant to quantum attacks than their asymmetric counterparts. This is because the time complexity of Grover's algorithm scales with the square root of the key size, meaning that doubling the key length results in a quantum-resistant symmetric encryption system. Therefore, the study of S-boxes, vectorial Boolean functions, and their equivalences remains crucial in the era of quantum computing.

This thesis attempts to explore and understand several methods of approaching computational methods for finding cryptographic functions that have been introduced in recent articles. We extend the theory and proofs in more mathematical theory that is usually left out of these works. We then implement algorithms and explain them in the context of the theory presented.

In Chapter 1 we give an introduction to the definitions and notations from algebraic theory and Boolean functions that will be used in the thesis.

In Chapter 2 we introduce the first method for searching quadratic APN functions using recursive search tree based on [2, Section 3]. We also use two EA-invariants to classify found functions up to EA-equivalence, which we then use to implement an algorithm for such classification.

In Chapter 3 we introduce a method for finding representatives for linear-equivalence classes for vectorial Boolean function in a given dimension, using [2, Section 4], [3, Section 4] and module theory from [4, Chapters 10 and 12]. Such representatives are tuples of matrices in rational canonical form. Based on the theory, we present an algorithm for finding such tuples.

In Chapter 4 we introduce the theory and algorithm of function trimming based on [5, Section 3], which for a given APN function in dimension n can find APN functions in dimension $n-1$.

In Chapter 5 we introduce the theory and algorithm for finding quadratic APN functions with maximum linearity based on [5, Section 5]. Using a quadratic APN function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$, we can construct a quadratic APN function with maximum linearity from $\mathbb{F}_2^{n+1}$ to $\mathbb{F}_2^{n+1}$.

1. Notation and Preliminaries

In this chapter we will introduce the most important definitions, notations, theorems and propositions. These statements will be necessary for the following chapters, where we will prove more complex theorems based on them. Most of the content of this chapter is based on [6], [5, Section 2] and on the [4, Chapter 1].

1.1 General Assumptions

This section is dedicated to the notation which is used throughout the thesis. We use the notation := in the sense of "define" to define a new element/set/etc.

We assume that $n \in \mathbb{N}$, where $\mathbb{N}$ is the set of all positive integers, $\mathbb{F}_q$ denotes the finite field with q elements ($q \in \mathbb{N}$, such that $q = p^n$, where p is prime), I_n denotes the identity matrix of dimension n, 0 denotes zero element, thus zero vector, zero matrix and zero number (depending on the context), and $\mathcal{L}(\mathbb{F}_2^n, \mathbb{F}_2^m)$ denotes all linear mappings from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$. Note that any $L \in \mathcal{L}(\mathbb{F}_2^n, \mathbb{F}_2^m)$ can be represented by matrix with n columns and m rows.

For an $n \times m$ matrix A we denote the transpose of the matrix A as A^T. We consider the elements $y \in \mathbb{F}_2^n$ to be column vectors, so y^T is a row vector. We only distinguish between column and row vectors when we use this vector in operations where the form is important, i.e. matrix multiplication. We will sometimes use the notation $a \oplus b$ for $a, b \in \mathbb{F}_2^n$, $a = (a_1, \dots, a_n)$, $b = (b_1, \dots, b_n)$, where $a \oplus b = (a_1 + b_1, \dots, a_n + b_n)$ and the addition is in $\mathbb{F}_2$ to indicate that we are working on elements from $\mathbb{F}_2^n$. Also for $u, x \in \mathbb{F}_2^n$ we will denote the monomial $\prod_{i=1}^n x_i^{u_i}$ by x^u.

We denote by $\mathrm{GL}(n, \mathbb{F}_2)$ the group of $n \times n$ invertible matrices over $\mathbb{F}_2$ and $\mathrm{AGL}(n, \mathbb{F}_2)$ denote the group of affine bijections on $\mathbb{F}_2^n$. Note also that an affine function $A : \mathbb{F}_2^n \to \mathbb{F}_2^m$ has the form

$$A(x) = L(x) + c,$$

where $x \in \mathbb{F}_2^n$, $L \in \mathcal{L}(\mathbb{F}_2^n, \mathbb{F}_2^m)$ and c is a constant from $\mathbb{F}_2^m$. For $n, m, p, q \in \mathbb{N}$ and matrices A, B over a field $\mathbb{F}$ such that

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}, \qquad B = \begin{pmatrix} b_{11} & \cdots & b_{1q} \\ \vdots & \ddots & \vdots \\ b_{p1} & \cdots & b_{pq} \end{pmatrix}$$

we have a defined $\mathrm{diag}(A, B)$ as

$$\mathrm{diag}(A, B) := \begin{pmatrix} a_{11} & \cdots & a_{1n} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & b_{11} & \cdots & b_{1q} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & b_{p1} & \cdots & b_{pq} \end{pmatrix}.$$

Let us consider a finite set $\mathcal{A}$. We will denote the cardinality of this set as $|\mathcal{A}|$. We will also use the notation $[a]$, where a is a positive real number, to denote the integral part of a, therefore $[a] := \max\{m \in \mathbb{Z} \mid m \leq x\}$.

1.2 Algebraic Theory

In this thesis we assume that the following terms are familiar to the reader and do not need to be defined in this thesis. Such terms are group, abelian group, ring, principal ideal domain (P.I.D. for short) and field. However, the definitions are given in [4, Chapter 10].

Definition 1. *A mapping* $\langle \cdot, \cdot \rangle : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2$ *is a* symmetric bilinear form *if it satisfies*

1. $\langle u, v \rangle = \langle v, u \rangle$ *for all* $u, v \in \mathbb{F}_2^n$,
2. $\langle u, v + w \rangle = \langle u, v \rangle + \langle u, w \rangle$ *for all* $u, v, w \in \mathbb{F}_2^n$,
3. $\langle \alpha u, v \rangle = \alpha \langle u, v \rangle$ *for all* $\alpha \in \mathbb{F}_2$ *and for all* $u, v \in \mathbb{F}_2^n$.

Definition 2. *The* inner product *is a symmetric bilinear form such that*

$$\begin{gathered} \langle \cdot, \cdot \rangle : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2 \\ (u_1, \ldots, u_n), (v_1, \ldots, v_n) \mapsto u_1v_1 + u_2v_2 + \cdots + u_nv_n. \end{gathered}$$

Definition 3. *The* Hamming weight *of the vector* $u \in \mathbb{F}_2^n$, $u = (u_1, \ldots, u_n)$ *is the number of* u_i *such that* $u_i = 1$, *where* $i \in \{1, \ldots, n\}$. *We will denote the Hamming weight of the vector* u *as* $\mathrm{wt_H}(u)$.

Definition 4. *Let* $u = (u_1, \ldots, u_n) \in \mathbb{F}_2^n$ *and* $v = (v_1, \ldots, v_n) \in \mathbb{F}_2^n$. *We say that* u is covered by v, *denoted by* $u \preceq v$, *if these two vectors satisfy* $u_i \leq v_i$ $\forall i \in \{1, \ldots, n\}$.

Now we prove two remarks about the relation from the definition above.

Remark. Let $u = (u_1, \ldots, u_n) \in \mathbb{F}_2^n$, $v = (v_1, \ldots, v_n) \in \mathbb{F}_2^n$ and $w = (w_1, \ldots, w_n) \in \mathbb{F}_2^n$. Let $u \preceq v$ and $v \preceq w$. Then $u \preceq w$.

Proof. Since $u \preceq v$, we know that for all $i \in \{1, \ldots, n\}$ it holds that $u_i \leq v_i$. Also since $v \preceq w$, we know that for all $i \in \{1, \ldots, n\}$ it holds that $v_i \leq w_i$. Therefore for all $i \in \{1, \ldots, n\}$ it holds, that $u_i \leq v_i \leq w_i \implies u_i \leq w_i \implies u \preceq w$. $\square$

Remark. Let $u = (u_1, \ldots, u_n) \in \mathbb{F}_2^n$, $v = (v_1, \ldots, v_n) \in \mathbb{F}_2^n$ and $w = (w_1, \ldots, w_n) \in \mathbb{F}_2^n$. Let $u \preceq v$ and $u \preceq w$. Then neither $v \preceq w$ nor $w \preceq v$ may be true.

Proof. Let us use an example. Let us assume that

$$u = (0\ 0\ 1\ 0), \quad v = (0\ 0\ 1\ 1), \quad w = (1\ 0\ 1\ 0).$$

These vectors satisfy the conditions $u \preceq v$ and $u \preceq w$. But because for the first and last position in the vectors v and w we have neither $v \preceq w$ nor $w \preceq v$. $\square$

1.2.1 Modules

The first algebraic structure, which we will define, is the module. Such a structure is somewhat similar to the vector space, but with the difference that it suffices to work over a ring instead of a field. The definitions in this subsection can be found in [4, Chapter 10].

Definition 5. *Let R be a ring. A* left R-module *or a* left module over R *is a set M together with a binary operation $+$ on M under which M is an abelian group, and an action of R on M (that is, a map $R \times M \to M$) denoted by rm, for all $r \in R$ and for all $m \in M$ which satisfies*

1. *(r+s)m=rm+sm, for all $r, s \in R, m \in M$,*
2. *(rs)m=r(sm), for all $r, s \in R, m \in M$, and*
3. *r(m+n)=rm+rn, for all $r \in R$, $m, n \in M$.*

If the ring R has an identity element 1, we state the additional axiom:

4. *$1m = m$, for all $m \in M$.*

We will use the term "R-module M" because we only consider "left modules" and want to emphasise the set M. For the definition of the vector space it is sufficient to consider R as a field. This immediately gives us the observation that modules over a field F and vector spaces over F are the same. Since we are working with some group M, we can also restrict to some subgroup N of M, therefore we define the next three terms.

Definition 6. *Let R be a ring and let M be an R-module. An* R-submodule *of M is a subgroup N of M which is closed under the action of ring elements, i.e., $rn \in N$, for all $r \in R$, $n \in N$.*

Definition 7. *Let M be a R-module and let $N_1, \dots, N_n$ be submodules of M. The* sum *of $N_1, \dots, N_n$ is the set of all finite sums of elements from the sets N_i such that*

$$\{a_1 + a_2 + \cdots + a_n \mid a_i \in N_i, i \in \{1, \cdots, n\}\}.$$

We will denote this sum by $N_1 + \cdots + N_n$.

Definition 8. *A submodule N of M (possibly $N = M$) is* cyclic *if there exists an element $a \in M$ such that $N = Ra$, that is, if N is generated by one element:*

$$N = Ra = \{ra \mid r \in R\}.$$

Naturally, since we have algebraic structures, we can define mappings between them. Thus, we define the following.

Definition 9. *Let R be a ring and let M and N be R-modules. A map $\varphi : M \to N$ is an* R-module homomorphism *if it respects the R-module structures of M and N, i.e.,*

1. *$\varphi(x + y) = \varphi(x) + \varphi(y)$, for all $x, y \in M$ and*
2. *$\varphi(rx) = r\varphi(x)$, for all $r \in R$, $x \in M$.*

An R-module homomorphism is an isomorphism *(*of R-modules*) if it is both injective and surjective. The modules M and N are said to be* isomorphic*, denoted $M \cong N$, if there is some R-module isomorphism $\varphi : M \to N$.*

If $\varphi : M \to N$ is and R-module homomorphism, let $\mathrm{Ker}(\varphi) = \{m \in M \mid \varphi(m) = 0\}$ (the kernel *of φ) and let $\varphi(M) = \{n \in N \mid n = \varphi(m) \text{ for some } m \in M\}$ (the* image *of φ).*

Note that $\mathrm{Ker}(\varphi)$ from the definition is a submodule of M. Since the kernel is a subgroup of M, it suffices to show that $ra \in \mathrm{Ker}(\varphi)$ holds for arbitrary $r \in R$ and $a \in \mathrm{Ker}(\varphi)$. Hence, $\varphi(ra) = r\varphi(a) = r0 = 0$, which implies $ra \in \mathrm{Ker}(\varphi)$.

Now let us state a first isomorphism theorem for modules. The theorem can be found in [4, Section 10.2].

Theorem 1. *Let M, N be R-modules and let $\varphi : M \to N$ be an R-module homomorphism. Then $\mathrm{Ker}(\varphi)$ is a submodule of M and $M/\mathrm{Ker}(\varphi) \cong \varphi(M)$.*

Definition 10. *An R-module M is said to be* free *on the subset A of M if for every nonzero element $m \in M$, there exist unique nonzero elements $r_1, r_2, \dots, r_n$ of R and unique $a_1, a_2, \dots, a_n$ in A such that*

$$m = r_1a_1 + r_2a_2 + \cdots + r_na_n,$$

for some $n \in \mathbb{N} \cup \{0\}$. In this situation we say A is a basis *or* set of free generators *for F. If R is commutative ring the cardinality of A is called the* rank *of M.*

Note that the R-module M can be finitely generated but not free. This can happen because we require the uniqueness of $r_1, \dots, r_n \in R$ and this property does not need to be provided in the module M. We will see this property in the Proposition 5 for the $\mathbb{F}_2[x]$-module.

Definition 11. *Let $k \in \mathbb{N}$ and $M_1, \dots, M_k$ be a collection of R-modules. The collection of k-tuples $(m_1, m_2, \dots, m_k)$, where $m_i \in M_i$ with addition and action of R defined component-wise is called the* direct product *of $M_1, \dots, M_k$, denoted $M_1 \times M_2 \times \cdots \times M_k$.*

Note that the direct product of $M_1, \dots, M_k$ is also called the *direct sum* of $M_1, \dots, M_k$ and is denoted as $M_1 \oplus \cdots \oplus M_k$. In the thesis we will sometimes work with the direct sum of k copies of R, where $k \in \mathbb{N}$. We will denote this as $R^k := R \oplus R \oplus \cdots \oplus R$ (k times). The action of R is defined component-wise. This means that for $r \in R$ and $m = (m_1, \dots, m_k) \in M_1 \oplus \cdots \oplus M_k$ we have

$$rm = (rm_1, rm_2, \dots, rm_k).$$

Note that rm belongs to the direct sum of the M_i modules, since the element rm_i in each component is in the module M_i.

Now we make a remark about the relation between the sum of modules from the Definition 7, the uniqueness of the basis of modules and the direct sum of modules from the definition above. This will be useful later. The following statement is based on the Proposition 5 from [4, Section 10.3]. The proof is described in more detail in this thesis.

Proposition 2. *Let $N_1, \dots, N_k$, where $k \in \mathbb{N}$, be submodules of R-module M. If every $x \in N_1 + \dots + N_k$ can be written uniquely in the form $a_1 + \dots + a_k$, where $a_i \in N_i$, then the map*

$$\begin{aligned} \pi : N_1 + N_2 + \dots + N_k &\to N_1 \times N_2 \times \dots \times N_k \\ \pi(a_1 + a_2 + \dots + a_k) &= (a_1, a_2, \dots, a_k) \end{aligned}$$

is an isomorphism (of R-modules), therefore

$$N_1 + N_2 + \dots + N_k \cong N_1 \times N_2 \times \dots \times N_k.$$

Proof. First we show that the given map π is homomorphism. Suppose we have two elements $a_1 + \dots + a_k$ and $b_1 + \dots + b_k$ from $N_1 + \dots + N_k$ such that $a_i, b_i \in N_i$ for $i \in \{1, \dots, k\}$. Hence

$$\begin{aligned} \pi(a_1 + \dots + a_k + b_1 + \dots + b_k) &= \pi\left((a_1 + b_1) + (a_2 + b_2) + \dots + (a_k + b_k)\right) \\ &= (a_1 + b_1, a_2 + b_2, \dots, a_k + b_k) \\ &= (a_1, a_2 \dots, a_k) + (b_1, b_2, \dots, b_k), \end{aligned}$$

where the last equation is given by the definition of the direct product (sum) of modules, where the addition is defined component-wise. Now assume that we have some $r \in R$. Then

$$\begin{aligned} \pi\left(r(a_1 + a_2 + \dots + a_k)\right) &= \pi(ra_1 + ra_2 + \dots + ra_k) \\ &= (ra_1, ra_2, \dots, ra_k) \\ &= r(a_1, a_2, \dots, a_k), \end{aligned}$$

where we have the last equation from the action of the element R being defined component-wise.

Now we need to prove that the mapping π is surjective and injective. For the surjection we choose any $a \in N_1 \times N_2 \times \dots \times N_k$. This means, that there are $a_i \in N_i$ such that $a = (a_1, a_2, \dots, a_k)$, therefore if we define $\alpha = a_1 + a_2 + \dots + a_k$, we get $\alpha \in N_1 + \dots + N_k$ such that $\pi(\alpha) = a$.

For the injection let us assume that there exist $a, b \in N_1 + \dots + N_k$, $a \neq b$, such that $\pi(a) = \pi(b)$. We know that

$$\begin{aligned} a &= a_1 + a_2 + \dots + a_k \text{ for some unique } a_i \in N_i, \\ b &= b_1 + b_2 + \dots + b_k \text{ for some unique } b_i \in N_i. \end{aligned}$$

The sums for a and b are unique, and since we assume that $a \neq b$, then there exist $j \in \{1, \dots, k\}$ such that $a_j \neq b_j$, which implies that $\pi(a)$ and $\pi(b)$ differ in the j-th coordinate. But this contradicts the assumption that $\pi(a) = \pi(b)$.

□

Assuming that the elements can be written uniquely, the following proposition allows us to better examine the structure of the direct sum of modules. The statement is based on the Proposition 3 from [4, Section 10.3]. The proof is described in more detail in this thesis.

Proposition 3. *Let $N_1, N_2, \ldots, N_k$ be submodules of R-module M. Then if every $x \in N_1 + \cdots + N_k$ can be written uniquely in the form $a_1 + a_2 + \cdots + a_k$ with $a_i \in N_i$, then*

$$N_j \cap (N_1 \times N_2 \times \cdots \times N_{j-1} \times N_{j+1} \times \cdots \times N_k) = 0 \text{ for all } j \in \{1, \ldots, k\}.$$

Proof. Since every x can be written uniquely, we can use the Proposition 2. Thus we have the isomorphism $\pi : N_1 \times N_2 \times \cdots \times N_k \to N_1 + N_2 + \cdots + N_k$.

For contradiction, let us assume that we have $a_j \in (N_1 \times \cdots \times N_{j-1} \times N_{j+1} \times \cdots \times N_k) \cap N_j$ such that $a_j \neq 0$. We can restrict the isomorphism π by excluding the submodule N_j. This is possible due to the fact, that each element x can be written uniquely, so elements with j-th coordinate equal to zero can also be written uniquely. Therefore we can apply the Proposition 2 and we get

$$N_1 \times \cdots \times N_{j-1} \times N_{j+1} \times \cdots \times N_k \simeq N_1 + \cdots + N_{j-1} + N_{j+1} + \cdots + N_k.$$

Using this isomorphism we know that $a_j = a_1 + \cdots + a_{j-1} + a_{j+1} + \cdots + a_k$. Thus $0 = a_1 + \cdots + a_{j-1} - a_j + a_{j+1} + \cdots + a_k$. Thus

$$\begin{aligned}
(\underbrace{0, \ldots, 0}_{n \text{ times}}) &= \pi^{-1}(0) \\
&= \pi^{-1}(a_1 + \cdots + a_{j-1} - a_j + a_{j+1} + \cdots + a_k) \\
&= (a_1, \ldots, a_{j-1}, -a_j, a_{j+1}, \ldots, a_k).
\end{aligned}$$

Which is a contradiction to π being an isomorphism, since we have a nonzero element $(a_1, \ldots, a_{j-1}, -a_j, a_{j+1}, \ldots, a_k) \in \mathrm{Ker}(\pi)$.

□

Definition 12. *An element m of the R-module M is called a* torsion element *if $rm = 0$ for some nonzero element $r \in R$. The set of torsion elements is denoted*

$$\mathrm{Tor}(M) = \{m \in M \mid rm = 0 \text{ for some nonzero } r \in R\}.$$

1.2.2 Modules over Polynomial Ring

In this subsection we will bring the theory from the previous subsection into the scope of this thesis. The most used algebraic structure will be for us a finite field $\mathbb{F}_2$. Over this finite field we will consider an n-dimensional vector space $\mathbb{F}_2^n$. This vector space satisfies all the properties of the definition of an R-module (Definition 5), thus we can understand the $\mathbb{F}_2^n$ as a $\mathbb{F}_2$-module $\mathbb{F}_2^n$.

Another useful observation is that we can also consider the $\mathbb{F}_2^n$ vector space over $\mathbb{F}_2$ as a $\mathbb{F}_2[x]$-module $\mathbb{F}_2^n$, since $\mathbb{F}_2[x]$ is a polynomial ring. This needs some explanation, as the observation is not very straightforward. It follows [4, Section 10.1] but we put it in the scope of the $\mathbb{F}_2[x]$-module $\mathbb{F}_2^n$.

First we have to choose an arbitrary linear transformation T from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$. This transformation is mandatory for defining the action from the definition of the R-module. Note that any linear transformation can be expressed in the matrix form. Thus, applying this transformation to the vector $v \in \mathbb{F}_2^n$ can be understood as multiplying the vector v by the matrix representing such a transformation from the left side. Also, since we are considering linear transformations, we have the following property for two linear transformations $A, B : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and two elements $\alpha, \beta \in \mathbb{F}_2$:

$$(\alpha A + \beta B)(v) = \alpha(A(v)) + \beta(B(v)).$$

For the linear transformation T we define

$$\begin{aligned} T^0 &= I, \\ &\vdots \\ T^k &= T \circ T \circ \cdots \circ T \quad (k \text{ times}), \end{aligned}$$

where I is the identity map of $\mathbb{F}_2^n \to \mathbb{F}_2^n$, $\circ$ denotes the composition of the mappings, and $k \in \mathbb{N} \cup \{0\}$.

Now we define the action of a polynomial in x on $\mathbb{F}_2^n$. Let $p(x)$ be an arbitrary polynomial in $\mathbb{F}_2[x]$ such that

$$p(x) = a_k x^k + a_{k-1}x^{k-1} + \cdots + a_1 x + a_0.$$

Then for $v \in \mathbb{F}_2^n$ we define an action of the ring elements $p(x)$ on the module element v by

$$\begin{aligned} p(x)v &= \left(a_k T^k + a_{k-1}T^{k-1} + \cdots + a_0\right)(v) \\ &= a_k T^k(v) + a_{k-1}T^{k-1}(v) + \cdots + a_1 T(v) + a_0 v. \end{aligned}$$

From this definition of the action we can say that x acts on $\mathbb{F}_2^n$ as the linear transformation T. We can observe that this definition of the action on $\mathbb{F}_2^n$ is consistent with the action of the field $\mathbb{F}_2$ on the vector space $\mathbb{F}_2^n$, since the field $\mathbb{F}_2$ is a subring of $\mathbb{F}_2[x]$ as constant polynomials. Thus our definition of the action $\mathbb{F}_2$ extends to the action on $\mathbb{F}_2[x]$.

Now we make an observation about $\mathrm{Tor}(\mathbb{F}_2^n)$. Since we have defined the action of x on $\mathbb{F}_2^n$, we can now observe $\mathrm{Tor}(\mathbb{F}_2^n)$. From the Definition 12 we have

$$\mathrm{Tor}(\mathbb{F}_2^n) = \{v \in \mathbb{F}_2^n \mid pv = 0 \text{ for some nonzero } p \in \mathbb{F}_2[x]\}.$$

Since all elements of $\mathbb{F}_2[x]$ are of the form

$$p(x) = a_k x^k + a_{k-1}x^{k-1} + \cdots + a_1 x + a_0,$$

where $k \in \mathbb{N} \cup \{0\}$. We can follow the previous observation and therefore choose any linear transformation $T : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and vector $v \in \mathbb{F}_2^n$, thus

$$p(x)v = a_k T^k(v) + a_{k-1}T^{k-1}(v) + \cdots + a_1 T(v) + a_0 v.$$

The above equation can be understood as a linear combination of elements from $\mathbb{F}_2^n$. These elements are $T^k(v), T^{k-1}(v), \cdots, T(v), v$ and the coefficients $a_k, a_{k-1}, \ldots, a_1, a_0 \in \mathbb{F}_2$. Since we know that $\mathbb{F}_2^n$ is a finitely generated vector space of dimension n, then any sequence of $n+1$ elements from $\mathbb{F}_2^n$ is linearly dependent, thus for any $v \in \mathbb{F}_2^n$ there exist $b_n, \ldots, b_0 \in \mathbb{F}_2$ such that

$$b_n T^n(v) + b_{n-1}T^{n-1}(v) + \cdots + b_1 T(v) + b_0 v = 0.$$

If we denote $r(x) = b_n x^n + b_{n-1}x^{n-1} + \cdots + b_1 x + b_0$, then $r(x) \in \mathbb{F}_2[x]$ and $r(x)v = 0$, which implies that v is a torsion element, and since our choice of v was arbitrary, we get that $\mathbb{F}_2^n = \mathrm{Tor}(\mathbb{F}_2^n)$.

As mentioned earlier, some modules can be finitely generated but not free. This is the case of the module $\mathbb{F}_2^n$. The vector space $\mathbb{F}_2^n$ is finitely generated over $\mathbb{F}_2$, thus it is finitely generated over $\mathbb{F}_2[x]$ using the action of elements from $\mathbb{F}_2[x]$ on $\mathbb{F}_2^n$. But while the $\mathbb{F}_2$-module $\mathbb{F}_2^n$ is free, the $\mathbb{F}_2[x]$-module $\mathbb{F}_2^n$ is not free. We will prove this in the following statements.

The following proposition is general, as we will need it later for a module other than the $\mathbb{F}_2$-module $\mathbb{F}_2^n$. It is based on the theory from [4, Section 10.1].

Proposition 4. *Let R be a ring with identity and let $m \in \mathbb{N}$. Then R-module R^n is free module of rank n over R.*

Proof. We assume that R contains identity, so we can define the set $A = \{e_1, e_2, \ldots, e_n\}$, $e_i \in R^n$ for $i \in \{1, \ldots, n\}$, where e_i consists of identity on the i-th component and zero on the j-th component for all $j \neq i$.

Consider any nonzero element $v \in R^n$ such that $v = (v_1, \ldots, v_n)$, where $v_i \in R$. Then we can express v using elements from the set A. Thus

$$v = v_1 e_1 + v_2 e_2 + \cdots + v_n e_n.$$

This gives us a unique expression of the v, since the i-th component of v is only affected by the i-th summand $v_i e_i$, so the i-th component is equal to v_i. Therefore we found the basis A with cardinality n.

□

Proposition 5. $\mathbb{F}_2[x]$*-module* $\mathbb{F}_2^n$ *is not free.*

Proof. Suppose for the sake of contradiction that $\mathbb{F}_2^n$ is free as a $\mathbb{F}_2[x]$-module. Then there exist basis $b_1, \ldots, b_k \in \mathbb{F}_2^n$, $k \in \mathbb{N}$ such that $\forall v \in \mathbb{F}_2^n$, $v \neq 0$,

$$v = r_1(x)b_1 + r_2(x)b_2 + \cdots + r_k(x)b_k, \tag{1.1}$$

where $r_1, \ldots, r_k \in \mathbb{F}_2[x]$ are unique nonzero elements. Since $\mathrm{Tor}(\mathbb{F}_2^n) = \mathbb{F}_2^n$, we know that $\forall b_i$, $i \in \{1, \ldots, k\}$ there exist nonzero $p_{b_i}(x) \in \mathbb{F}_2[x]$ such that $p_{b_i}(x)b_i = 0$. Therefore

$$\begin{aligned} r_1(x)b_1 + r_2(x)b_2 + \cdots + r_k(x)b_k &= v - 0 \\ &= v - \left(p_{b_1}(x)b_1 + p_{b_2}(x)b_2 + \cdots + p_{b_k}(x)b_k\right). \end{aligned}$$

Thus

$$v = \left(r_1(x) + p_{b_1}(x)\right) b_1 + \left(r_2(x) + p_{b_2}(x)\right) b_2 + \cdots + \left(r_k(x) + p_{b_k}(x)\right) b_k.$$

Now we have two possible cases. If there is $j \in \{1, \ldots, k\}$ such that $r_j(x) \neq p_{b_j}(x)$, then $r_j(x) + p_{b_j}(x) \neq 0$, which is contradicts the uniqueness of the expression of v in the equation 1.1.

The second case is when for all $i \in \{1, \ldots, k\}$ we have $r_i(x) = p_{b_i}(x)$. This implies that $r_i(x) + p_{b_i}(x) = 0$ and therefore $v = 0$, which contradicts with the choice of v as a nonzero element.

□

1.2.3 Finite Field

Another important algebraic structure for this thesis is the finite field $\mathbb{F}_{2^n}$ and its relation to the vector space $\mathbb{F}_2^n$. We are interested in the structure of the elements of the finite field $\mathbb{F}_{2^n}$. Note that $\mathbb{F}_{2^n}$ is an abstract notion for a finite field with 2^n elements, and that such a finite field exists, since 2 is prime and we assume that $n \in \mathbb{N}$.

We start with the finite field $\mathbb{F}_2$. This finite field is isomorphic to $\mathbb{Z}_2$, since all finite fields with the same number of elements are isomorphic. According to the theory of splitting fields and finite fields [7, Chapter 28], there exists an element α which is a root of an irreducible polynomial of degree n over $\mathbb{Z}_2$. Let us denote such a polynomial by $m_{\mathbb{Z}_2,\alpha}$. This polynomial forms a finite field $\mathbb{Z}_2[x]/m_{\mathbb{Z}_2,\alpha}$ with 2^n elements. This finite field is isomorphic to $\mathbb{Z}_2(\alpha)$, since they both have 2^n elements and two finite fields with the same amount of elements are isomorphic [7, Chapter 28]. Thus we have an isomorphism between finite fields $\mathbb{F}_{2^n}$ and $\mathbb{Z}_2(\alpha)$. Elements of $\mathbb{Z}_2(\alpha)$ are of the form

$$\sum_{i=0}^{n-1} a_i \alpha^i,$$

where $a_i \in \mathbb{F}_2$. Therefore, we can think of $\mathbb{Z}_2(\alpha)$ as a vector space over $\mathbb{Z}_2$. Since this vector space is finite, it has a finite basis. For example, such a basis is $\{1, \alpha, \alpha^2, \dots, \alpha^{n-1}\}$.

Now we have a look at the vector space $\mathbb{F}_2^n$. This vector space is also finite and has for example the basis $e_1, \dots, e_n$, where $e_i \in \mathbb{F}_2^n$ is a zero vector with one on the i-th coordinate, where $i \in \{1, \dots, n\}$. Therefore we can introduce a mapping φ of vector spaces such that

$$\begin{aligned} \varphi : \mathbb{Z}_2(\alpha) &\to \mathbb{F}_2^n \\ \alpha^i &\mapsto e_{i+1}, \end{aligned}$$

where $i \in \{0, \dots, n-1\}$. This mapping is an isomorphism of the vector spaces $\mathbb{Z}_2(\alpha)$ and $\mathbb{F}_2^n$, thus we have that $\mathbb{F}_{2^n}$ and $\mathbb{F}_2^n$ are isomorphic as vector spaces. This implies that the vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ can be understood as a function from $\mathbb{F}_{2^n}$ to $\mathbb{F}_{2^n}$.

In the thesis (usually in algorithms) we will denote vectors of $\mathbb{F}_2^n$ by non-negative integers, because each vector of $\mathbb{F}_2^n$ corresponds to the binary notation of a non-negative integer $a \in \{0, \dots, 2^n - 1\}$.

1.3 Introduction to Boolean Functions

Definition 13. *A* Boolean function in dimension *n is a function from the n-dimensional vector space $\mathbb{F}_2^n$ to $\mathbb{F}_2$.*

We usually do not include the dimension when we use the term *Boolean function*, or we can say *Boolean function in n variables*, since the n is the number of input bits in the Boolean function.

Definition 14. *Let $f_1, \dots, f_m$ be Boolean functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2$. The function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ given by $F(x) = (f_1(x), \dots, f_m(x))$, where $x \in \mathbb{F}_2^n$, is called* vectorial Boolean function. *The functions $f_i(x)$, where $i \in \{1, \dots, m\}$, are called* coordinate functions *of F.*

1.3.1 Representation

From [6] or [8, Section 2.2.1] we know that any Boolean function f can be uniquely expressed in *Algebraic Normal Form* (ANF for short) as follows

$$f(x) = \sum_{u \in \mathbb{F}_2^n} a_u \prod_{i=1}^{n} x_i^{u_i},$$

where $a_u \in \mathbb{F}_2$, $x = (x_1, \ldots, x_n)$, $u = (u_1, \ldots, u_n)$. And similarly for the vectorial Boolean function F

$$F(x) = \sum_{u \in \mathbb{F}_2^n} a_u \prod_{i=1}^{n} x_i^{u_i},$$

where $a_u \in \mathbb{F}_2^n$, $x = (x_1, \ldots, x_n)$, $u = (u_1, \ldots, u_n)$.

We will also represent functions in the form of look-up tables. For the vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, where $m \in \mathbb{N}$, the look-up table is an array with 2^n positions denoted from 0 to $2^n - 1$. This denotation represents the vector of $\mathbb{F}_2^n$ as an input to the vectorial Boolean function. Thus, at the i-th position we have the value $F(i)$.

1.3.2 Properties

Definition 15. *A vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is called* almost perfect nonlinear *(APN for short) if*

$$F(x) + F(x + a) = b$$

has at most 2 solutions for all $a \neq 0 \in \mathbb{F}_2^n$, $b \in \mathbb{F}_2^m$.

One of the most important properties for us is that the vectorial Boolean function can be APN. This property gives us the ability to deal with the (non)linearity of the Boolean function. Since linearity is a bad property for functions to have in terms of linear cryptanalysis, we are interested in those that are almost non-linear.

The reason, why the following definition works with two solutions and not just one is very much simple. Suppose we have a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and we have found some $x \in \mathbb{F}_2^n$ such that for all $a \neq 0$, $a \in \mathbb{F}_2^n$ and for all $b \in \mathbb{F}_2^n$ satisfies $F(x) + F(x + a) = b$. Now let us take some $y = x + a$. This gives us the following

$$F(y) + F(y + a) = F(x + a) + F(x + a + a) = F(x + a) + F(x) = b.$$

This implies that whenever we have found a solution x of the equation $F(x) + F(x + a) = b$, we know that that also $y = x + a$ is also a solution of this equation, so the minimum number of solutions (if any solution exists) is two, since $a \neq 0$.

Definition 16. *Let $\alpha \in \mathbb{F}_2^n \backslash \{0\}$ and $\beta \in \mathbb{F}_2^n$. Let F be a vectorial Boolean function such that $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. The* Difference Distribution Table *(DDT for short) is a two-dimensional array in which the rows correspond to all possible values of α and the columns correspond to all possible values of β. The element in the α-th row and the β-th column will then be*

$$\mathrm{DDT}^F_{\alpha,\beta} = \left|\{x \in \mathbb{F}_2^n \mid F(x) + F(x + \alpha) = \beta\}\right|.$$

The DDT of the vectorial Boolean function F is useful to decide whether the function F is APN or not. It is enough to look in the table, and if in any position there is a number greater than 2, then the function F is not APN. Because of the property we mentioned after the definition of APN, we know that the values in DDT are even numbers. This property can be used in some algorithms to find APN functions by evaluating DDT for the function. If the value in DDT is greater than 2, then the function cannot be APN.

Using the DDT we can define the differential spectrum. This spectrum will have an impact in the Chapter 2 where we will prove that such a spectrum is invariant under a certain equivalence relation of vectorial Boolean functions.

Definition 17. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a vectorial Boolean function. The* differential spectrum *is the multiset of the values* $\mathrm{DDT}^F_{\alpha,\beta}$.

Definition 18. *The* algebraic degree *of a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ with ANF*

$$f(x) = \sum_{u \in \mathbb{F}_2^n} a_u \prod_{i=1}^{n} x_i^{u_i},$$

where $a_u \in \mathbb{F}_2$, is defined as

$$\deg(f) = \max\{\mathrm{wt_H}(u) : u \in \mathbb{F}_2^n \mid a_u \neq 0\}.$$

The algebraic degree *of a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is defined as the maximum algebraic degree over all of its coordinate functions.*

Functions with an algebraic degree of 2 *are called* quadratic*. Functions with an algebraic degree of* 1 *or less are called* affine*.*

Definition 19. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a vectorial Boolean function. A* component *of F is a function $\mathbb{F}_2^n \to \mathbb{F}_2$, $x \mapsto \langle b, F(x)\rangle$, where $b \in \mathbb{F}_2^n \setminus \{0\}$. The* Walsh transform *of F at the point $(\alpha, \beta) \in \mathbb{F}_2^n \times (\mathbb{F}_2^n \setminus \{0\})$ is defined as*

$$\widehat{F}_\beta(\alpha) := \sum_{x \in \mathbb{F}_2^n} (-1)^{\langle \alpha, x\rangle + \langle \beta, F(x)\rangle}$$

and the linearity *of F corresponds to the maximum absolute value of its Walsh transform, i.e.,*

$$\max_{\alpha \in \mathbb{F}_2^n, \beta \in \mathbb{F}_2^n \setminus \{0\}} \left| \widehat{F}_\beta(\alpha) \right|.$$

We have given the definition for vectorial Boolean functions, but the Walsh transform and linearity can also be defined for Boolean functions. In this case, in the definition of the Walsh transform, we can replace the component $\langle \beta, F(x)\rangle$ with the value of the Boolean function for the entry x. In the definition of linearity, we search for the maximum only over all $\alpha \in \mathbb{F}_2^n$, since we have no β.

We can put the values of Walsh transform into a two-dimensional array, thus we define the following.

Definition 20. *Let $\alpha \in \mathbb{F}_2^n$ and $\beta \in \mathbb{F}_2^n \setminus \{0\}$. Let F be a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. The* Walsh spectrum*, denoted by $\mathcal{W}_F$, is a two-dimensional array whose rows correspond to all possible values of α and whose columns correspond to all possible values of β. The element in the α-th row and the β-th column is $\widehat{F}_\beta(\alpha)$. Therefore*

$$\mathcal{W}_F = \left\{ \widehat{F}_\beta(\alpha) \mid \alpha \in \mathbb{F}_2^n, \beta \in \mathbb{F}_2^n \setminus \{0\} \right\}.$$

Definition 21. *Let $n \in \mathbb{N}$, $n > 2$ and $\mathbb{F}_2^n$. We say that a quadratic APN function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ has* maximum linearity *if it has linearity 2^{n-1}.*

As with the differential spectrum, we can define the following spectrum, which is another invariant under a certain equivalence relation of vectorial Boolean functions, as we will prove in Chapter 2.

Definition 22. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be vectorial Boolean function. The multiset*

$$\mathcal{F}_F = \left\{ \left| \widehat{F_\beta}(\alpha) \right| \mid \alpha \in \mathbb{F}_2^n, \beta \in \mathbb{F}_2^n \setminus \{0\} \right\}$$

is called the extended Walsh spectrum.

1.4 Equivalence Relations

In this section we will introduce four well-known equivalence relations. These are *linear equivalence*, *affine equivalence*, *extended-affine equivalence* and *Carlet-Charpin-Zinoviev equivalence* (CCZ-equivalence for short). We will also mention important relations between these equivalence relations.

Definition 23. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and $G : \mathbb{F}_2^n \to \mathbb{F}_2^m$, where $n, m \in \mathbb{N}$, be two vectorial Boolean functions. We call these two functions F and G* linear-equivalent*, if there exist a $\mathbb{F}_2$-linear automorphism of $\mathbb{F}_2^n$ L and a $\mathbb{F}_2$-linear automorphism of $\mathbb{F}_2^m$ L' such that*

$$G = L' \circ F \circ L.$$

Definition 24. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and $G : \mathbb{F}_2^n \to \mathbb{F}_2^m$, where $n, m \in \mathbb{N}$, be two vectorial Boolean functions. We call these two functions F, G* extended-affine equivalent *(*EA-equivalent *for short, denoted $F \approx_{\mathrm{EA}} G$) if there exist affine bijections $A_1 : \mathbb{F}_2^n \to \mathbb{F}_2^n$, $A_2 : \mathbb{F}_2^m \to \mathbb{F}_2^m$ and an affine function $B : \mathbb{F}_2^n \to \mathbb{F}_2^m$ such that*

$$G = A_2 \circ F \circ A_1 + B.$$

Moreover, if B is a zero matrix, we call the vectorial Boolean functions F and G affine-equivalent*.*

Proposition 6. *The relation EA-equivalence is indeed an equivalence. In other words, the relation EA-equivalence is reflexive, symmetric and transitive.*

Proof. Let us assume that we have vectorial Boolean functions, affine bijections and affine function as stated in the Definition 24. First we prove reflexivity. Suppose we have given a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$. We want to find affine bijections A_1, A_2 and affine function B such that $\forall x \in \mathbb{F}_2^n$ it holds that

$$F(x) = A_2 \circ F \circ A_1(x) + B(x).$$

It is sufficient to choose $A_1 = I_m$, $A_2 = I_n$ and B as a zero matrix $m \times n$.

Now for symmetry. Suppose we have $G = A_2 \circ F \circ A_1 + B$, where $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $G : \mathbb{F}_2^n \to \mathbb{F}_2^m$ are vectorial Boolean functions, $A_1 : \mathbb{F}_2^n \to \mathbb{F}_2^n$, $A_2 : \mathbb{F}_2^m \to \mathbb{F}_2^m$ are affine bijections and $B : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is an affine function. Since A_1, A_2

are bijections, we know that there exist their inversions, which are also affine bijections. Therefore we can rewrite F as

$$F = A_2^{-1} \circ G \circ A_1^{-1} + A_2^{-1} \circ B \circ A_1^{-1},$$

where A_2^{-1} and A_1^{-1} are affine bijections and $A_2^{-1} \circ B \circ A_1^{-1}$ is an affine function since A_2^{-1} and A_1^{-1} are affine bijections.

Finally, we show transitivity. Suppose F, G, H are vectorial Boolean functions such that $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $G : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $H : \mathbb{F}_2^n \to \mathbb{F}_2^m$ such that $F \approx_{\text{EA}} G$ and $G \approx_{\text{EA}} H$, therefore

$$G = A_2 \circ F \circ A_1 + B_G, \qquad H = A_4 \circ G \circ A_3 + B_H,$$

where $A_1 : \mathbb{F}_2^n \to \mathbb{F}_2^n$, $A_2 : \mathbb{F}_2^m \to \mathbb{F}_2^m$, $A_3 : \mathbb{F}_2^n \to \mathbb{F}_2^n$, $A_4 : \mathbb{F}_2^m \to \mathbb{F}_2^m$ are affine bijections and $B_G : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $B_H : \mathbb{F}_2^n \to \mathbb{F}_2^m$ are affine functions.

We want to show that then $F \approx_{\text{EA}} H$. Therefore

$$\begin{aligned} H &= A_4 \circ G \circ A_3 + B_H \\ &= A_4 \circ (A_2 \circ F \circ A_1 + B_G) \circ A_3 + B_H \\ &= A_4 \circ (A_2 \circ F \circ A_1 \circ A_3 + B_G \circ A_3) + B_H. \end{aligned}$$

Now we have to rewrite A_4 and since A_4 is an affine bijection, it is of the form $A_4 = L_4 + v$, where L_4 is a linear bijective mapping from $\mathbb{F}_2^m$ to $\mathbb{F}_2^m$ and v is a vector in $\mathbb{F}_2^m$. Thus

$$\begin{aligned} &= L_4 (A_2 \circ F \circ A_1 \circ A_3 + B_G \circ A_3) + v + B_H \\ &= L_4 \circ A_2 \circ F \circ A_1 \circ A_3 + L_4 \circ B_G \circ A_3 + v + B_H. \end{aligned}$$

Now let‘s look at $L_4 \circ A_2 \circ F \circ A_1 \circ A_3$. $L_4 \circ A_2$ is an affine bijection because L_2 is linear bijection and A_2 is an affine bijection. $A_1 \circ A_3$ is also an affine bijection since A_1 and A_3 are affine bijections.

The remainder is $L_4 \circ B_G \circ A_3 + v + B_H$. $L_4 \circ B_G \circ A_3$ is an affine mapping since L_4 is linear mapping and B_G and A_3 are affine mappings. Also $v + B_H$ is an affine mapping since B_H is an affine function. Thus we have a sum of affine mappings which is an affine mapping and therefore $F \approx_{\text{EA}} H$.

□

To define CCZ-equivalence, we need to formally define a graph of a function.

Definition 25. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. We denote by Γ_F the* graph of the function F

$$\Gamma_F = \{(x, F(x)) \mid x \in \mathbb{F}_2^n\} \subset \mathbb{F}_2^{2n}.$$

Definition 26. *Let F, $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$. The functions F and G are* CCZ-equivalent *if there exists an affine permutation $\sigma : \mathbb{F}_2^{2n} \to \mathbb{F}_2^{2n}$ such that $\sigma(\Gamma_F) = \Gamma_G$.*

We can now establish relations between these equivalence relations.

Proposition 7. *The relationship between linear equivalence, affine equivalence, EA-equivalence and CCZ-equivalence is as follows*

$$\textit{linear equiv.} \implies \textit{affine equiv.} \implies \textit{EA-equiv.} \implies \textit{CCZ-equiv.}$$

Proof. We can see the first two implications by expanding the relation of two functions satisfying $F \approx_{\mathrm{EA}} G$. Suppose we have functions F, G and A_1, A_2, B as in the definition of EA-equivalence (Definition 24). Let us assume, that $A_1 = L_1 + c_1$, $A_2 = L_2 + c_2$ and $B = L_B + c_B$, where $L_1 \in \mathcal{L}(\mathbb{F}_2^n, \mathbb{F}_2^n)$, $L_2 \in \mathcal{L}(\mathbb{F}_2^m, \mathbb{F}_2^m)$ and $L_B \in \mathcal{L}(\mathbb{F}_2^n, \mathbb{F}_2^m)$ are bijections, $c_1 \in \mathbb{F}_2^n$ and $c_2, c_B \in \mathbb{F}_2^m$. Then for any $x \in \mathbb{F}_2^n$ we have

$$\begin{aligned} G(x) &= (A_2 \circ F \circ A_1)(x) + B(x) \\ &= A_2 \circ F\left(L_1(x) + c_1\right) + L_B(x) + c_B \\ &= L_2\left(F\left(L_1(x) + c_1\right)\right) + c_2 + L_B(x) + c_B. \end{aligned}$$

If we now choose $c_1 = 0$, $c_2 = 0$, $c_B = 0$ and $L_B = 0$, then we get $G(x) = L_2(F(L_1(x)))$, which satisfies definition of linear equivalence. If we keep $L_B = 0$ and $c_B = 0$, but we let $c_1 \in \mathbb{F}_2^n$ and $c_2 \in \mathbb{F}_2^m$, we get $G(x) = L_2\left(F(L_1(x) + c_1)\right) + c_2$, which satisfies definition of affine equivalence. And finally, if we let $L_B \in \mathcal{L}(\mathbb{F}_2^n, \mathbb{F}_2^m)$ and $c_B \in \mathbb{F}_2^m$, then we have a formula that satisfies EA-equivalence.

For the last implication we assume that we have $\Gamma_F = \{(x, F(x)) \mid x \in \mathbb{F}_2^n\}$, $\Gamma_G = \{(y, G(y)) \mid y \in \mathbb{F}_2^n\}$ and that $F \approx_{\mathrm{EA}} G$. We want to show that there exists an affine permutation σ such that $\Gamma_G = \sigma(\Gamma_F)$, in other words, we want σ such that for any $y \in \mathbb{F}_2^n$ we have some $x \in \mathbb{F}_2^n$ which satisfies

$$\begin{pmatrix} y \\ G(y) \end{pmatrix} = \sigma\left(\begin{pmatrix} x \\ F(x) \end{pmatrix}\right).$$

From the definition of EA-equivalence we have

$$G(x) = A_2 \circ F \circ A_1(x) + B(x).$$

This implies that

$$G \circ A_1^{-1}(x) = A_2 \circ F(x) + B \circ A_1^{-1}(x).$$

If we assume that $A_1(x) = L_1(x) + l_1$, $A_2(x) = L_2(x) + l_2$ and $B(x) = L_B(x) + l_B$, where $L_1, L_2 \in \mathrm{GL}(n, \mathbb{F}_2)$, $L_B \in \mathcal{L}(\mathbb{F}_2^n, \mathbb{F}_2^m)$ and $l_1, l_2, l_B \in \mathbb{F}_2^n$, we can write

$$A_2 \circ F(x) + B \circ A_1^{-1}(x) = L_2\left(F(x)\right) + l_2 + L_B L_1^{-1} x + L_B L_1^{-1} l_1 + l_B,$$

thus if we use a substitution $y = A_1^{-1}(x)$ we get

$$\begin{aligned} \begin{pmatrix} y \\ G(y) \end{pmatrix} &= \begin{pmatrix} A_1^{-1}(x) \\ A_2 \circ F(x) + B \circ A_1^{-1}(x) \end{pmatrix} \\ &= \begin{pmatrix} L_1^{-1} x + L_1^{-1} l_1 \\ L_2 F(x) + l_2 + L_B L_1^{-1} x + L_B L_1^{-1} l_1 + l_B \end{pmatrix} \\ &= \begin{pmatrix} L_1^{-1} x & 0 \\ L_B L_1^{-1} & L_2 \end{pmatrix} \begin{pmatrix} x \\ F(x) \end{pmatrix} + \begin{pmatrix} L_1^{-1} l_1 \\ L_B L_1^{-1} l_1 + l_2 + l_B \end{pmatrix}, \end{aligned}$$

which means that we have found the σ.

$\square$

Our primary focus is on APN functions and EA-equivalence. We therefore state the following proposition, which establishes a strong link between these two concepts. This is an important contribution to the future chapters of this thesis.

Proposition 8. *Let F, G be two vectorial Boolean functions $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $G : \mathbb{F}_2^n \to \mathbb{F}_2^m$ such that $F \approx_{\mathrm{EA}} G$. Then function F is APN if and only if G is APN.*

Proof. For the first implication, let us assume that G is APN. From the definition of EA-equivalence we know, that $G = A_1 \circ F \circ A_2(x) + B(x)$, where $A_1 : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $A_2 : \mathbb{F}_2^m \to \mathbb{F}_2^m$ are affine bijections and $B : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is an affine function. Since A_2 is an affine bijection, it is of the form $A_2(x) = L_2(x) + c_2$, where $L_2 : \mathbb{F}_2^m \to \mathbb{F}_2^m$ is a linear bijection and $c_2 \in \mathbb{F}_2^m$.

Let us choose an arbitrary $a \in \mathbb{F}_2^n$, $a \neq 0$. We define $D_G^a := \{G(x)+G(x+a) \mid x \in \mathbb{F}_2^n\}$. Since we assume that G is APN, we know that any value $G(x)+G(x+a)$ will appear in the set with multiplicity at most two. We are interested in the multiplicity of the elements of the set, thus we rewrite the condition of the set.

$$\begin{aligned} D_G^a &= \{G(x) + G(x+a) \mid x \in \mathbb{F}_2^n\} \\ &= \{A_1 \circ F \circ A_2(x) + B(x) + A_1 \circ F \circ A_2(x+a) + B(x+a) \mid x \in \mathbb{F}_2^n\} \end{aligned}$$

The function B is affine, thus $B(x+a) = B(x) + B(a)$, therefore

$$= \{A_1 \circ F \circ A_2(x) + A_1 \circ F \circ A_2(x+a) + B(a) \mid x \in \mathbb{F}_2^n\}.$$

$B(a)$ is a constant, therefore it does not affect the multiplicity of the elements.

$$\begin{aligned} &= \{A_1 \circ (F \circ A_2(x) + F \circ A_2(x+a)) \mid x \in \mathbb{F}_2^n\} \\ &= \{A_1 \circ (F \circ (L_2(x) + c_2) + F \circ (L_2(x+a) + c_2)) \mid x \in \mathbb{F}_2^n\} \\ &= \{A_1 \circ (F \circ (L_2(x) + c_2) + F \circ (L_2(x) + L_2(a) + c_2)) \mid x \in \mathbb{F}_2^n\} \end{aligned}$$

The mapping A_1 is an affine bijection, therefore it does not change the multiplicity of elements in the set. We also define $y := L_2(x) + c_2$. Thus

$$D_G^a = \{F(y) + F(y + L_2(a)) \mid x \in \mathbb{F}_2^n\} = D_F^{L_2(a)}.$$

We get, that $D_G^a = D_F^{L_2(a)}$. This means that the elements in the set $D_F^{L_2(a)}$ have a multiplicity of at most 2, so F is APN. For the second implication (assuming that F is APN) we can use the same proof.

□

Now we give the theorem about the relation between EA-equivalence and CCZ-equivalence for quadratic APN vectorial Boolean functions. The theorem and proof can be found in [9].

Theorem 9. *Let $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be two quadratic APN vectorial Boolean functions with $n \geq 2$. Then F is CCZ-equivalent to G if and only if F is EA-equivalent to G.*

2. Recursive Tree Search and Classifying up to EA-equivalence

This chapter introduces the first method which can possibly find new vectorial Boolean APN functions. This method uses a recursive tree search. The number of all possible vectorial Boolean functions grows exponentially, since for a dimension n we have $(2^n)^{2^n}$ of possible choices. Therefore, we need to define some constraints on the search. The whole chapter is based on the [2, Section 3].

We begin with a section introducing and proving a method for computing the coefficients of the ANF form of a vectorial Boolean function. This method helps us to find quadratic APN vectorial Boolean functions. These found functions can be pairwise EA-equivalent, so in the next section we prove that the differential spectrum and the extended Walsh spectrum are invariants under EA-equivalence, so we can use them for classifying functions into EA-equivalence classes. The last section is devoted to the algorithm of the recursive tree search for quadratic APN vectorial Boolean functions and to the algorithm which places these functions in EA-equivalence classes.

Our contribution is to present the theory in a clearer way, to prove the statements in more detail, to present a corrected implementation of the algorithm from [2, Section 3] for searching a quadratic APN vectorial Boolean functions and describe it in the context of the theory from this chapter, to prove that the differential spectrum and the extended Walsh spectrum are EA-invariants, and to implement our own algorithm for classifying found functions up to EA-equivalence using these EA-invariants.

2.1 Coefficients in ANF

In this section we will use the information provided in [10, Section 2] and [8, Section 2.2]. The goal is to find a simple way to calculate the coefficients of the ANF of the vectorial Boolean function. At first, we will concentrate only on the Boolean functions. We will show, that it is possible to compute the coefficient $a_u \in \mathbb{F}_2$ for $u \in \mathbb{F}_2^n$ from an ANF of a Boolean function f using values of f for $x \preceq u$. Then we will show, that the same is possible for a vectorial Boolean function. The following lemma and proof follows the theory from [8, Section 2.2.1].

Lemma 10. *Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a Boolean function. Let $f(x) = \sum_{u \in \mathbb{F}_2^n} a_u x^u$ be ANF of f. Then*

$$f(x) = \bigoplus_{u \preceq x} a_u. \tag{2.1}$$

Proof. We have ANF

$$f(x) = \sum_{u \in \mathbb{F}_2^n} a_u x^u.$$

Since this is a sum of elements in $\mathbb{F}_2^n$, we can rewrite it as

$$f(x) = \bigoplus_{u \in \mathbb{F}_2^n} a_u x^u.$$

If $x^u = 0$, then also $a_u x^u = 0$, which means that these elements do not affect the sum. Therefore, we are only interested in the case where $x^u = 1$. We extend the notation of the monomial, thus $1 = x^u = \prod_{i=1}^n x_i^{u_i}$. Since both u_i and x_i are from $\mathbb{F}_2$, there are only four possible cases. If $x_i = 0$ and $u_i = 1$, then $x_i^{u_i} = 0 \implies x^u = 0$. In all other cases ($x_i = 0$, $u_i = 0$ and $x_i = 1$, $u_i \in \mathbb{F}_2$) we have $x_i^{u_i} = 1$. Therefore, if $u \preceq x$ we have $x^u = 1$. This means that we can rewrite the Boolean function f as $f(x) = \bigoplus_{u \preceq x} a_u$.

□

Using the Equation 2.1, we can prove the following statement, which gives us a simple method for computing the coefficients a_u just by using the values of $f(x)$ for $x \preceq u$. The statement and proof follow Theorem 1 [8, Section 2.2].

Proposition 11. *Let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a Boolean function. Let $f(x) = \bigoplus_{u \in \mathbb{F}_2^n} a_u x^u$ be ANF form of f. Then we have for each $u \in \mathbb{F}_2^n$ that*

$$a_u = \bigoplus_{x \preceq u} f(x)$$

Proof. Let us start by defining $b_u := \bigoplus_{x \preceq u} f(x)$. With this b_u, we can consider a new function $g(x)$, which we can define as follows

$$g(x) := \bigoplus_{u \in \mathbb{F}_2^n} b_u x^u.$$

Using the Lemma 10, we can rewrite $g(x)$ as

$$g(x) = \bigoplus_{u \preceq x} b_u.$$

If we expand b_u, we get

$$g(x) = \bigoplus_{u \preceq x} \left(\bigoplus_{y \preceq u} f(y) \right).$$

Now we want to transform these two xors so that they depend on the variable y. To do this, we fix y. We want to know how often the value $f(y)$ appears in those two xors of $g(x)$. Let $u_1, \ldots, u_k$ be all such vectors, that satisfy $y \preceq u_i$ and $u_i \preceq x$. Thus $f(y)$ appears in the xors of $g(x)$ exactly k times. This implies, that we can rewrite $g(x)$ as follows

$$g(x) = \bigoplus_{y \in \mathbb{F}_2^n} f(y) \left(\bigoplus_{y \preceq u \preceq x} 1 \right).$$

We will examine the second xor. If $y \preceq x$ and $y \neq x$, then $\mathrm{wt_H}(y) \leq \mathrm{wt_H}(x)$ must hold. Furthermore, since $y \neq x$, $\mathrm{wt_H}(y) < \mathrm{wt_H}(x)$ must hold. Thus, due to $y \preceq u \preceq x$, the number of positions in the vector u that can be either 0 or 1, is equal to $\mathrm{wt_H}(y) - \mathrm{wt_H}(u)$ (the elements on the remaining positions are fixed). This implies, that the number of v satisfying $u \preceq v \preceq x$ is equal to

$$2^{\mathrm{wt_H}(x) - \mathrm{wt_H}(u)},$$

which is an even number. This implies that we have a xor of an even number of ones, so the xor is equal to zero. If neither $x \preceq y$ nor $y \preceq x$ is true, or if $x \preceq y$, then we have no possible choice for u.

The remaining case is $y = x$. Then we have only one possible choice for u and that is $u = y$, so the second xor is equal to one. All together we have that the second xor is equal to zero, except for the case when $y = x$. This implies that $g(x) = f(y)$ for $y = x$, so $g(x) = f(x)$.

Since we know, that the ANF is unique, it must hold that $a_u = b_u$ for all $u \in \mathbb{F}_2^n$. Therefore, $\forall u \in \mathbb{F}_2^n$ we have that $a_u = \oplus_{x \preceq u} f(x)$.

□

Therefore we know, that for an arbitrary Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ with an ANF $f(x) = \sum_{u \in \mathbb{F}_2^n} a_u x^u$, we can compute a_u as $a_u = \bigoplus_{x \preceq u} f(x)$.

We know, that any vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ can be expressed as $F(x) = (f_1(x), \dots, f_m(x))$, where $f_i(x) : \mathbb{F}_2^n \to \mathbb{F}_2$ are coordinate (Boolean) functions of F, for $i \in \{1, \dots, n\}$. We can therefore prove the following proposition.

Proposition 12. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $F(x) = (f_1(x), \dots, f_m(x))$ be a vectorial Boolean function with ANF such that $F(x) = \sum_{u \in \mathbb{F}_2^n} a_u x^u$, where $a_u \in \mathbb{F}_2^m$. Then for all $u \in \mathbb{F}_2^n$*

$$a_u = \bigoplus_{x \preceq u} F(x).$$

Proof. Let us fix some arbitrary $u \in \mathbb{F}_2^n$. We want to compute a_u. The element a_u is an m-dimensional vector. Now for all $i \in \{1, \dots, m\}$, let us take i-th coordinate from the vector, which we will denote $a_{u,i}$. The i-th coordinate of the function F is determined by Boolean function $f_i(x)$. Therefore, for $a_{u,i} \in \mathbb{F}_2$, we can use Proposition 11 and we get $a_{u,i} = \bigoplus_{x \preceq u} f_i(x)$. Thus,

$$a_u = \begin{pmatrix} a_{u,1} \\ a_{u,2} \\ \vdots \\ a_{u,m} \end{pmatrix} = \begin{pmatrix} \bigoplus_{x \preceq u} f_1(x) \\ \bigoplus_{x \preceq u} f_2(x) \\ \vdots \\ \bigoplus_{x \preceq u} f_m(x) \end{pmatrix} = \bigoplus_{x \preceq u} F(x)$$

□

Proposition 13. *Let $n \geq 3$, $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, $F(x) = (f_1(x), \dots, f_m(x))$ be a vectorial Boolean function with ANF $F(x) = \sum_{u \in \mathbb{F}_2^n} a_u x^u$, where $a_u \in \mathbb{F}_2^m$. The function $F(x)$ is quadratic if and only if for all $u \in \mathbb{F}_2^n$, which satisfies $\mathrm{wt}_{\mathrm{H}}(u) \geq 3$, $a_u = 0$.*

Proof. The proof follows directly from the uniqueness of the ANF of the vectorial Boolean function, where a_u is the coefficient of the monomial x^u.

□

2.2 EA-invariants

In this section we prove that the differential spectrum and the extended Walsh spectrum are EA-invariants. We will use this property in the following section where we present an algorithm for classifying quadratic APN vectorial Boolean functions up to EA-equivalence. The fact that these spectra are EA-invariants is stated without proof in [2, Section 3].

Theorem 14. *Let $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be two vectorial Boolean functions such that $F \approx_{\mathrm{EA}} G$. Then differential spectrum of F consists of the same elements with the same multiplicity as differential spectrum of G, i.e. differential spectrum is EA-invariant.*

Proof. Since $F \approx_{\mathrm{EA}} G$, we know that there exist affine bijections $A_1, A_2 : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and affine function $B : \mathbb{F}_2^n \to \mathbb{F}_2^n$ such that $G = A_2 \circ F \circ A_1 + B$. Let us take an element $a \in \mathbb{N}$ from the differential spectrum of G. This means that for some fixed $\alpha \in \mathbb{F}_2^n \setminus \{0\}, \beta \in \mathbb{F}_2^n$ there exist $x_1, \ldots, x_a$ such that $G(x_i) + G(x_i + \alpha) = \beta$ for $i \in \{1, \ldots, a\}$. Thus

$$\begin{aligned}
\beta &= G(x_i) + G(x_i + \alpha) \\
\beta &= A_2 \circ F \circ A_1(x_i) + B(x_i) + A_2 \circ F \circ A_1(x_i + \alpha) + B(x_i + \alpha) \\
\beta &= A_2 \circ F \circ A_1(x_i) + A_2 \circ F \circ A_1(x_i + \alpha) + B(\alpha) \\
\beta &= A_2 \circ F \circ A_1(x_i) + A_2 \circ F \left(A_1(x_i) + A_1(\alpha)\right) + B(\alpha).
\end{aligned}$$

If we denote $y_i := A_1(x_i)$, then we get

$$\begin{aligned}
\beta &= A_2 \circ F(y_i) + A_2 \circ F\left(y + A_1(\alpha)\right) + B(\alpha) \\
A_2^{-1}(\beta) &= A_2^{-1}\left(A_2 \circ F(y_i) + A_2 \circ F\left(y_i + A_1(\alpha)\right) + B(\alpha)\right) \\
A_2^{-1}(\beta) &= F(y_i) + F\left(y_i + A_1(\alpha)\right) + A_2^{-1}\left(B(\alpha)\right)
\end{aligned}$$

and therefore

$$F(y_i) + F\left(y_i + A_1(\alpha)\right) = A_2^{-1}\left(\beta + B(\alpha)\right).$$

This implies that for $\alpha' := A_1(\alpha)$ and $\beta' := A_2^{-1}\left(\beta + B(\alpha)\right)$ we have that

$$F(y_i) + F\left(y_i + \alpha'\right) = \beta'.$$

Since this is true for all $x_i \in \{x_1, \ldots, x_a\}$ and A_1 is an affine bijection, we have that a belongs to the differential spectrum of F and the multiplicity of a in the differential spectrum of F is equal to the multiplicity of a in the differential spectrum of G.

$\square$

Theorem 15. *Let $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be two vectorial Boolean functions such that F is EA-equivalent to G. Then $\mathcal{F}_F = \mathcal{F}_G$, i.e. the extended Walsh spectrum is EA-invariant.*

Proof. We start with notation where x^T, for $x \in \mathbb{F}_2^n$ is row vector, thus for $y \in \mathbb{F}_2^n$ we can write $\langle x, y \rangle = x^T y$.

Since $F \approx_{\mathrm{EA}} G$, we know that the function $G = A_1 \circ F \circ A_2 + A_3$ where A_1, A_2 are affine bijections, so they are in a form $A_i = L_i + c_i$ for $i \in \{1, 2\}$, where L_i is linear bijection and $c_i \in \mathbb{F}_2^n$ and A_3 is affine function, which is of the form $A_3 = L_3 + c_3$, where L_3 is linear function and $c_3 \in \mathbb{F}_2^n$. We want to prove that its possible to rewrite $\widehat{G_\gamma}(\delta)$, where $\gamma \in \mathbb{F}_2^n$ and $\delta \in \mathbb{F}_2^n \setminus \{0\}$, in a form as $\widehat{F_\alpha}(\beta)$. Therefore we want to find some specific α and β.

$$\widehat{G_\gamma}(\delta) = \sum_{x \in \mathbb{F}_2^n} (-1)^{\langle \delta, x\rangle + \langle \gamma, G(x)\rangle}$$

We will now focus on the exponent. Therefore we have

$$\langle \delta, x\rangle + \langle \gamma, G(x)\rangle = \delta^T x + \gamma^T \left(L_1 \circ F\left(L_2 x + c_2\right) + c_1 + L_3 x + c_3\right)$$

Now we denote $z = L_2 x + c_2$ which implies that $x = L_2^{-1} z + L_2^{-1} c_2$. Thus for the exponent

$$\begin{aligned}
&= \delta^T \left(L_2^{-1} z + L_2^{-1} c_2\right) + \gamma^T \left(L_1 F(z) + c_1 + L_3 L_2^{-1} z + L_3 L_2^{-1} c_2 + c_3\right) \\
&= \left(\delta^T L_2^{-1} + \gamma^T L_3 L_2^{-1}\right) z + \gamma^T L_1 F(z) + \left(\delta^T L_2^{-1} c_2 + \gamma^T c_1 + \gamma^T L_3 L_2^{-1} c_2 + \gamma^T c_3\right) \\
&= \left(\delta^T L_2^{-1} + \gamma^T L_3 L_2^{-1}\right) z + \gamma^T L_1 F(z) + \epsilon
\end{aligned}$$

where $\epsilon = \delta^T L_2^{-1} c_2 + \gamma^T c_1 + \gamma^T L_3 L_2^{-1} c_2 + \gamma^T c_3$ is a constant. Therefore

$$\begin{aligned}
&= \left(\left(\delta^T L_2^{-1}\right)^T + \left(\gamma^T L_3 L_2^{-1}\right)^T\right)^T z + \left(L_1^T \gamma\right)^T F(z) + \epsilon \\
&= \langle \alpha, z\rangle + \langle \beta, F(z)\rangle + \epsilon,
\end{aligned}$$

for $\alpha = \left(\delta^T L_2^{-1}\right)^T + \left(\gamma^T L_3 L_2^{-1}\right)^T$ and $\beta = L_1^T \gamma$. Thus we have

$$\widehat{G_\gamma}(\delta) = \sum_{x \in \mathbb{F}_2^n} (-1)^{\langle \delta, x\rangle + \langle \gamma, G(x)\rangle} = (-1)^\epsilon \sum_{z \in \mathbb{F}_2^n} (-1)^{\langle \alpha, z\rangle + \langle \beta, F(x)\rangle} = \pm \widehat{F_\beta}(\alpha)$$

This implies that $|\widehat{G_\gamma}(\delta)| = |\widehat{F_\alpha}(\beta)|$, which implies that extended Walsh spectra consists of the same elements with the same multiplicity.

□

2.3 Algorithm

In this section we will use the term "quadratic APN function" instead of the term "quadratic APN vectorial Boolean function". In the following subsections we gradually develop an algorithm for recursive tree search of a quadratic APN function. In the first subsection we present an algorithm that searches for a quadratic APN function for a given dimension. In the next subsection we modify the algorithm so that it searches for all quadratic APN functions in a given dimension or it tries to randomly generate quadratic APN functions for a given number of runs. In the last subsection we present an algorithm that put found quadratic APN functions to the EA-equivalence classes.

All of the algorithms are implemented in the Python programming language, but in this text we will use pseudo-code for better understanding. The implementations of algorithms and results are attached in A.

For clarity, we will use the notation, where names written in `This_Format` are functions, names written in "this_format" (including the quotes) are arrays, and names written in *this_format* are integers or real numbers. We will also refer to $array[x]$ as the x-th position in an array "array".

2.3.1 Search for a Quadratic APN Function

The core of the algorithm follows the algorithm from [2, Section 3], thus the functions `Add_Degree_Information`, `Remove_Degree_Information`, `Generate_P`, `Next_Val`, `Add_Point` and `Remove_Point` are taken from [2]. The functions `Add_DDT_Information` and `Remove_DDT_Information` are taken from [3, Section 5]. We will use these functions and describe them in more detail in this paper. However, we will make some changes to the algorithm, because the algorithm in [2] is not correct. This will be discussed later.

Description of the Algorithm

Our implementation of the following algorithm is attached in A.1.

Algorithm 1 APN search using recursive search tree

```
Output: a quadratic APN function
1: $n \leftarrow$ integer representing the dimension in which we want to search for APN function
2: $init\_value \leftarrow -1$
3: "output_sbox" $\leftarrow$ initialised as an empty array
4: "sbox" $\leftarrow$ Generate_Sbox($n$)
5: "P" $\leftarrow$ Generate_P($n$)
6: "DDT" $\leftarrow$ Generate_DDT($n$)
7: "SUM", "CTR" $\leftarrow$ Generate_SUM_and_CTR($n$)
8: Next_Val(0)
```

The lines 1-3 of the algorithm contain global variables and an array which are initialised to the desired default values. The variable *init_value* is set to the value -1, which is the value, that will not appear as output from any process in the algorithm. The variable "output_sboxes" will store all APN functions found. On the lines 4-7 more global arrays are initialised, but initialising each of them requires a function.

We start with the function `Generate_Sbox`. It creates an array "sbox" of size 2^n, which is initialised with a value init_value. The output of this function is therefore an array with the value *init_value* repeated 2^n-times.

The next function is `Generate_P`. Its purpose is to generate a global two-dimensional array "P" of size $2^n \times 2^n$ containing a list of 2^n permutations of $\{0, \ldots, 2^n - 1\}$. The randomness is provided by the *random* package in Python. Its purpose is in higher dimensions, since it is not feasible to search through all possible choices of the vectorial Boolean function (more on this later in Subsection

2.3.2). Therefore, we want to search through different vectorial Boolean functions in each run of the algorithm.

The third function is `Generate_DDT`. This function is used to generate a two-dimensional array “DDT” of size $2^n \times 2^n$. This array stores the DDT information of the vectorial Boolean function, i.e. the i-th row is assigned to α, which is equal to i, and the j-th column is assigned to β, which is equal to j, where α and β are taken from the definition of DDT. Therefore for some $x \in \mathbb{F}_2^n$ which satisfies $F(x) + F(x + \alpha) = \beta$, we increment the element $DDT[\alpha][\beta]$ by one. Using this array, we can check, during the recursive search, if the function can still be APN, or terminate the branch of the tree search and try another one.

The final initialisation function is the `Generate_SUM_and_CTR` function. Its purpose is to create 1×2^n arrays “SUM” and “CTR” with all positions initialised to zero. The array “SUM” stores information about the coefficients of the ANF calculated according to Proposition 12, and the array “CTR” stores the number of x already considered that satisfy $x \preceq u$, since the xor in Proposition 12 runs through all $x \preceq u$.

We will now describe the following two functions: `Hamming_Weight` and `Vector_Ordering`. The `Hamming_Weight` function calculates the Hamming weight of a given integer, which represents a binary notation as mentioned at the end of Subsection 1.2.3.

The function `Vector_Ordering` evaluates whether the given numbers a and b (represented as elements from $\mathbb{F}_2^n$) satisfy the relation $a \preceq b$ from the Definition 4. This relation is not satisfied if and only if there exists i such that $a_i = 1$ and $b_i = 0$.

Now we examine the function `Add_Point`. This function uses other two functions. Therefore we will describe them as well.

```
1: function ADD_POINT($x$):
2:     if Add_DDT_Information($x$) then
3:         return(Add_Degree_Information($x$))
4:     end if
5:     return(0)
6: end function
```

We start with the function `Add_DDT_Information`. Its purpose is to keep the array “DDT” updated, which means that when we call this function with the parameter x, we have fixed x and we want to update “DDT” with all the α and β from Definition 16. Therefore, if $F(x)+F(x+\alpha) = \beta$, we increase the element $DDT[\alpha][\beta]$ by two. This is because the algorithm assigns a value to the array “sbox” in an ascending order, which means that “sbox” has a value different from *init_value* for all positions in $[0, \dots, x]$. Thus, when this function checks if $sbox[x \oplus \alpha] \neq init\_value$, it can only happen if $0 \leq x \oplus \alpha < x$. Therefore, for all $x \oplus \alpha > x$ we have $sbox[x \oplus \alpha] = init\_value$. An important observation is that for every x that satisfies the equation from the definition of APN, there exists $x' = x \oplus \alpha$ that also satisfies the equation.

There are two possible cases. The first is that $x < x'$, thus when we examine x we get $sbox[x \oplus \alpha] = init\_value$, so we do not change $DDT[\alpha][\beta]$. But if we examine x' in another call of this function, we have $sbox[x'] \neq init\_value$, so we have to increase the $DDT[\alpha][\beta]$ by two (one for x and one for x'). During each DDT update, the function also checks (on the line 5), if the APN property is not violated. If so, it returns zero.

```
 1: function Add_DDT_Information($x$):
 2:     for $\alpha \in [1, \dots, 2^n]$ do
 3:         if $sbox[x \oplus \alpha] \neq init\_value$ then
 4:             $DDT[\alpha][sbox[x] \oplus sbox[x \oplus \alpha]] \leftarrow$ increase the value by 2
 5:             if $DDT[\alpha][sbox[x] \oplus sbox[x \oplus \alpha]] > 2$ then
 6:                 return(0)
 7:             end if
 8:         end if
 9:     end for
10:     return(1)
11: end function
```

The function `Add_Degree_Information` checks, if the searched function can still be quadratic during the recursive tree search. This is done by following Proposition 13. In the array "SUM" we keep the updated values of the coefficients a_u of ANF of the vectorial Boolean function searched with $\mathrm{wt_H}(u) \geq 3$. We apply Proposition 12 to the new x with the assignment value $sbox[x]$. On the line 4 we check if the condition $x \preceq u$ is fulfilled and if so, on the line 6 we add the value $sbox[x]$ to the sum, which is assigned to the coefficient a_u ($SUM[u]$). Then we check, if we consider all vectors x such that $x \preceq u$. The number of such vectors is exactly $2^{\mathrm{wt_H}(u)}$. If so, we check on the line 8, that the coefficient a_u is zero. If not, then the vectorial Boolean function examined cannot be quadratic.

```
 1: function Add_Degree_Information($x$):
 2:     for $u \in [1, \dots, 2^n]$ do
 3:         if Hamming_Weight($u$) $> 2$ then
 4:             if Vector_Ordering($x, u$) $= 1$ then
 5:                 $CTR[u] \leftarrow \mathrm{ctr}[u] + 1$
 6:                 $SUM[u] \leftarrow \mathrm{SUM}[u] \oplus \mathrm{sbox}[x]$
 7:                 if $CTR[u] = 2^{\texttt{Hamming\_Weight}(u)}$ then
 8:                     if $SUM[u] \neq 0$ then
 9:                         return(0)
10:                     end if
11:                 end if
12:             end if
13:         end if
14:     end for
15:     return(1)
16: end function
```

In summary, in the function `Add_Point`, the addition of the point x with the assigned value $sbox[x]$ can be done if the APN property is not violated (via function `Add_DDT_Information`) and if the function is still quadratic (via function `Add_Degree_Information`).

The last function of the algorithm, before we examine the function `Next_Val`, is the function `Remove_Point`, which again consists of two functions: `Remove_DDT_Information` and `Remove_Degree_Information`. These functions do the reverse of the functions `Add_DDT_Information` and `Add_Degree_Information`.

Finally, we can look at the function `Next_Val`. The function takes as an input the number x, which denotes the position in "sbox" (or the depth of the tree search) that we are interested in. The values for $sbox[x]$ are taken from the array "P", which contains a random permutation for each x. If this value does not violate the condition that the vectorial Boolean function is APN and quadratic (it passes the function `Add_Point` on the line 8), the function calls itself at the position $x + 1$. If one of the conditions is violated, or the function ends in the depth $x + 1$, the function calls the function `Remove_Point` on the line 12.

```
 1: function NEXT_VAL($x$):
 2:     if ISCOMPLETE(sbox) then
 3:         "output_sbox" $\leftarrow$ append "sbox" into the list
 4:         terminate all recursive searches
 5:     end if
 6:     for $z \in [0, \dots, 2^n - 1]$ do
 7:         $sbox[x] \leftarrow P[x][z]$
 8:         $b \leftarrow$ Add_Point($x$)
 9:         if $b = 1$ then
10:             Next_Val($x + 1$)
11:         end if
12:         Remove_Point($x$)
13:         $sbox[x] \leftarrow init\_value$
14:     end for
15: end function
```

Improvements and Changes

As mentioned above, in this subsection we will introduce some changes and improvements to the implementation of Algorithm 1 from [2]. We start with the changes.

The implementation of Algorithm 1 in [2] contains two more lines before the execution of the function `Next_Val` (lines 4 and 5 in the implementation from [2]). On the first one, the value 0 is assigned to the array "sbox" at the 0-th position, therefore the vectorial Boolean function we are looking for has a zero vector input and zero vector output. On the second, the function `Add_Point` is executed with the value 0.

The first one is removed in this thesis because it would restrict the search space for the vectorial Boolean functions, since the condition sets the value for the entire run of the algorithm and cannot be changed. Although the quadratic APN vectorial Boolean function does not have to have a zero vector as an output for a zero vector input. Since this line is removed in this thesis, the line with the function `Add_Point`(0) is also removed. It would make no sense to try to call this function, if we had not previously set the value $sbox[0]$.

Therefore, in the Algorithm 1 in this thesis, we try at position 0 in the array "sbox" all values from $[0, \dots, 2^n - 1]$.

Another change is made in the function `Next_Val`. We start with the comment on the line 11 in the Algorithm 1 from [2]. The comment says, that the function `Next_Free_Position` chooses the smallest i such that $sbox[i] \neq init\_value$. But as the algorithm is presented, this function would return 0 in every depth of the recursive search. Therefore the function `Next_Free_Position` should return the smallest i such that $sbox[i] = init\_value$. But as we mentioned, this would mean, that there is no way, that the algorithm would change the value $sbox[0]$ to $init\_value$, if we strictly follow the implementation from [2], because the function `Next_Free_Position` would never return $i = 0$.

The next change is to swap lines 19 and 20 of Algorithm 1 from [2]. The function `Remove_Point` contains the function `Remove_DDT_Information` which operates on the value $sbox[x]$. Thus, if we changed $sbox[x]$ to the value $init\_value$ first, we would get a nonsensical result.

Another change is, that we can replace the *depth* variable with x. The function fills the array "sbox" in an ascending order, and since the function is first called first with $x = 0$, there is no need for the *depth* variable.

The next change is that in the function `Next_Val` in this thesis, we completely remove the functions `Is_Complete` and `Next_Free_Position`. As already mentioned, we start from the 0-th position and the algorithm examines the positions in an ascending order, so the next position from x would always be $x+1$, therefore we do not need the function `Next_Free_Position`. Also, the array "sbox" has 2^n elements numbered from 0 to $2^n - 1$. Therefore, if $x = 2^n$, we know that the array does not contain the value $init\_value$ at any position.

The last changes are in the functions `Add_DDT_Information` and `Remove_DDT_Information`. In the [3] the functions do not return value 1. This means that in the functions `Add_Point` and `Remove_Point` the functions `Add_Degree_Information` and `Remove_Degree_Information` are never executed. Therefore we modify the functions `Add_DDT_Information` and `Remove_DDT_Information` by adding a line that returns 1 if the for-cycle was not interrupted. We also had to modify the functions by removing the line that checks if the Hamming weight of the α is even. This restriction only applies to the APN permutation, but since we are not only interested in the APN permutation, we need to remove this restriction.

Since finding a quadratic APN function may be infeasible in relatively small dimensions (starting at $n = 5$), we introduce a modification where the user can specify the number of runs and the number of seconds each run takes. In each such run, we reshuffle the array "P". This can improve our search, because since the algorithm is recursive, the choice of $sbox[x]$ for relatively small x can affect the search in such a way, that it is impossible to find the APN function with that choice of $sbox[x]$. Thus if we limit the search time with this choice, we can try another random permutation that might lead to the result. The implementation of the modified algorithm is attached in A.2.

Results

In dimension $n = 2, 3, 4, 5$ and 6 we are able to find quadratic APN function practically instantaneously. For dimension $n = 7$, we were unable to find a quadratic APN function after three attempts, each of which took 2 hours.

Using the modified algorithm we were able to find some quadratic APN functions for $n = 2, 3, 4, 5, 6, 7$. For $n = 8, 9$ we were unable to find quadratic APN functions after five tries, each for half an hour.

2.3.2 Search for Quadratic APN Functions

Description of the Algorithm

To have algorithm searching for all quadratic APN functions in given dimension we modify the Algorithm 1 in a way that in the function `Next_Val` we modify the line 4 that it only terminates the recursive iteration in which is the algorithm running in that moment. The implementation of the algorithm is attached in A.3.

Improvements and Changes

Again, since running the algorithm for $n = 4$ and larger can be infeasible, we modify the algorithm in similar way as the algorithm in the previous subsection. We can limit the time of each run and the number of runs. This will lead to the fact that we will not find all functions, but it can rise our chances to find at least some functions. If we modify the algorithm by adding more runs (in each we generate new array “P”), we can find some functions multiple time. Thus we use function `Remove_Duplicates` which return list of unique functions. The implementation of the modified algorithm is attached in A.4.

Results

For $n = 2$ we are able to find all 192 quadratic APN functions. For $n = 3$ we are able to find all 688128 quadratic APN functions, therefore those are all quadratic APN functions in these dimensions. This correspond to the found APN functions in [11, Table 4].

Using the modified algorithm we were able to find for 77014 quadratic APN functions for $n = 4$. For $n = 5$ we were able to find 28975 quadratic APN functions. For $n = 6$ we were able to find 13767 quadratic APN functions. For $n = 7$ we were able to find 2 quadratic APN functions. All of these results are attached in A.5.

2.3.3 Classifying Functions up to EA-equivalence

Description of the Algorithm

In the previous subsection we found some quadratic APN functions. Now we want to put them into EA-equivalence classes. Thus we introduce an algorithm that do that. The algorithm uses the theory from Section 2.2. The implementation of the algorithm is attached in A.6.

Algorithm 2 Classifying found quadratic APN vectorial Boolean functions up to EA-equivalence

```
Output: a list “classes” with classify functions to the EA-equivalence classes
1: $n \leftarrow$ integer representing the dimension in which we want to search for APN
   function
2: $init\_value \leftarrow -1$
3: “functions” $\leftarrow$ initialised with look-up tables of functions we want to sort
4: “classes” $\leftarrow$ Put_Functions_Into_EA_Classes(“functions”)
```

Before we examine the `Put_Functions_Into_EA_Classes` function, we need to describe the `Is_Complete` and `Next_Free_Position` functions. These functions are only mentioned in [2]. The `Is_Complete` function checks whether the given list contains the value *init_value* at any position. If so, the output of this function is 0. Otherwise, the output will be 1.

The next function is `Next_Free_Position`. It's purpose is to find the first position in the given list that contains the value *init_value* and give that position as an output. This function can be used in the `Put_Functions_Into_EA_Classes` function only after the `Is_Complete` function has returned 0, which means that there is a position in the list containing the value *init_value*.

Now we can focus on the function `Put_Functions_Into_EA_Classes`. It's purpose is to put the given array "functions" of quadratic APN vectorial Boolean functions into EA-equivalent classes. To indicate if the function is assigned to some EA-equivalent class, we use the array "is_in_some_class". The while loop checks if there is a function that is not in any EA-equivalence class. If there is such a function, it creates a new EA-equivalence class (array "EA_class") and appends this function to the class. Then the algorithm checks the EA-equivalence of the functions using `Can_Functions_Be_EA_equivalent`. This function computes extended Walsh spectra and differential spectra for the input functions and compares them according to Section 2.2. From the Chapter 5 of this thesis and Proposition 30 we can use the ortho-derivatives of the input functions and compute these spectra also for them. Therefore we get four EA-invariants. If all functions from the input array are put into some EA-equivalence class, the function returns the array "EA_classes" with the equivalence classes.

```
function Put_Functions_Into_EA_Classes("functions"):
    "EA_classes" ← initialised as an empty list
    "is_in_some_class" ← initialised as a list, which contains value init_value
  repeated k-times, where k is length of the array "functions"
    while Is_Complete("is_in_some_class")=0 do
        x ← Next_Free_Position("is_in_some_class")
        "EA_class" ← initialised as an empty list
        "EA_class" ← append element functions[x] into the list
        "F"←"functions[x]"
        "F_ortho"←ortho_derivative("F")
        is_in_some_class[x] ← 1
        for i ∈ [x + 1, ..., length of array "functions"] do
            if is_in_some_class[i] = init_value then
                "G"←"functions[i]"
                "G_ortho"←ortho_derivative("G")
                if (Can_Functions_Be_EA_equivalent("F","G")) and
(Can_Functions_Be_EA_equivalent("F_ortho","G_ortho")) then
                    "EA_class" ← append element "functions[i]" into the list
                    is_in_some_class[i] ← 1
                end if
            end if
        end for
        "EA_classes" ← append list "EA_class" into the list
    end while
```

```
24:     return(“EA_classes”)
25: end function
```

Improvements and Changes

Since we are using invariants, we cannot state with certainty that two functions are EA-equivalent if their invariants are equal. For example our Algorithm 2 put all found functions for $n = 5$ from Subsection 2.3.2 into one EA-equivalence class. This is in contradiction with the results from [11, Table 3].

These lead us to modification using theory and algorithm from [12] where the authors presented a method for finding matrices from the definition of EA-equivalence. This method is very time consuming, since the algorithm is trying to find the matrices from EA-equivalence. Even though we include the algorithm from [12] in our modified algorithm in a way that if the algorithm pass the condition on the line 16 we execute algorithm from [12]. Only if the algorithm from [12] finds matrices from EA-equivalence, we execute lines 17 and 18.

Only a few minor changes had to be made to the Python code `Equivalence.py` from [12]. All of these changes were made for the sole purpose of running the code. The changes are mentioned in the source code in A.7.

Results

$n = 2$

In Subsection 2.3.2 we have found 192 quadratic APN functions. Using the Algorithm 2 we put them all into one EA-equivalent class, which means that all of these functions are EA-equivalent.

$n = 3$

In Subsection 2.3.2 we have found 688128 quadratic APN functions. Using the Algorithm 2 we put them all into one EA-equivalent class, which means that all of these functions are EA-equivalent.

$n = 4$

In Subsection 2.3.2 we have found 77014 quadratic APN functions. Using the Algorithm 2 we put them all into one EA-equivalent class, which means that all of these functions are EA-equivalent.

$n = 5$

In Subsection 2.3.2 we have found 28975 quadratic APN functions. Using the modified Algorithm 2 we classified 2400 of them into 2 equivalence classes.

$n = 6$

In Subsection 2.3.2 we have found 13767 quadratic APN functions. Using the Algorithm 2 we put them into 13 equivalence classes.

$n = 7$

In Subsection 2.3.2 we have found 2 quadratic APN functions. Using the Algorithm 2 we sorted them into 2 equivalence classes. The two found functions are EA-equivalent to two EA-inequivalent functions found before in [13], which are listed in `sevenBitAPN.py` in [14].

The number of EA-equivalence classes for $n = 4$ and 5 corresponds to the number of EA-equivalence classes of quadratic APN functions in [11, Table 2, Table 3]. For $n = 6$ the number of EA-equivalence classes of quadratic APN functions corresponds to [15, Table 1], because of Theorem 9. The results for $n = 5, 6$ and 7 are attached in A.8.

3. LE-automorphism

This chapter is based on [2, Section 4], [3, Section 4] and [4, Chapters 10 and 12]. We will follow these papers as we extend the necessary theory and focus on explaining the process.

In this chapter we will consider a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. We will put such functions into linear-equivalence classes which admit a non-trivial linear self-equivalence (meaning that there exists a non-trivial element in its automorphism group). In the first section we introduce the automorphism group of the vectorial Boolean function F and its subgroup $\mathrm{Aut}_{\mathrm{LE}}$. Since the elements of this group are tuples of matrices, we will use module theory in the second section to show that every matrix is similar to a matrix in rational canonical form (RCF). Similarity is an equivalence relation, so it defines equivalence classes. The matrix in RCF is therefore representative of some equivalence class. In the third section we introduce the power-similarity of matrices, which helps us to extend the equivalence class for similarity to an equivalence class for power-similarity. Then, in the fourth section, we introduce an algorithm that classifies all functions that admit a non-trivial linear self-equivalence up to linear-equivalence in a given dimension.

Our contribution is to put the theory from [4, Chapter 12] into the context of the [2, Section 4]. Also, to prove the statements from [3, Section 4] in more detail, to present our own implementation of the algorithm for finding such linear-equivalence classes, which was approached purely on the basis of the theory presented and independently of the implementation attached to [2] and [3], and to provide results for dimensions up to 12.

3.1 Automorphism Group

In this section we will introduce the automorphism group and its associated equivalence relation using the similarity of matrices. The definitions are based on those from [3, Section 2] and [16, Section 5.1] respectively.

Definition 27. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$. We say that an affine permutation σ of $\mathbb{F}_2^{n+m}$, such that $\Gamma_F = \sigma(\Gamma_F)$, is* automorphism for F.

We can easily observe, that for a given function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$, all automorphisms for F form a group. The identity element is an automorphism I_{n+m} defined as $x \mapsto x$ for all elements of Γ_F. Associativity and the existence of an inverse element for any automorphism for F σ is satisfied since σ is an affine permutation.

Definition 28. *The* automorphism group *of a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is defined as*

$$\mathrm{Aut}(F) := \{\sigma \in \mathrm{AGL}(2n, \mathbb{F}_2) \mid \Gamma_F = \sigma(\Gamma_F)\}.$$

Now we use the restriction to the matrices of the affine permutations to get the following subgroup.

Definition 29. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. Then we define the finite subgroup* $\mathrm{Aut}_{\mathrm{LE}}(F)$ *as follows*

$$\mathrm{Aut}_{\mathrm{LE}}(F) := \left\{ \sigma \in \mathrm{Aut}(F) \mid \sigma = \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix} \text{ for } A, B \in \mathrm{GL}(n, \mathbb{F}_2) \right\}.$$

For any element $\sigma \in \mathrm{Aut}_{\mathrm{LE}}(F)$, such that $\sigma = \mathrm{diag}(A, B)$, where $A, B \in \mathrm{GL}(n, \mathbb{F}_2)$, we will use the notation (A, B), where (A, B) is called the tuple. An alternative way of expressing an element of $\mathrm{Aut}_{\mathrm{LE}}(F)$ is given by the following proposition.

Proposition 16. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. Then*

$$\begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix} \in \mathrm{Aut}_{\mathrm{LE}}(F) \iff F \circ A = B \circ F.$$

Proof. From the definition of $\mathrm{Aut}_{\mathrm{LE}}(F)$ (and $\mathrm{Aut}(F)$) we have that for some $x, y \in \mathbb{F}_2^n$

$$\begin{pmatrix} y \\ F(y) \end{pmatrix} = \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix} \begin{pmatrix} x \\ F(x) \end{pmatrix}.$$

This is true if and only if

$$y = Ax \quad \text{and} \quad F(y) = B \circ F(x).$$

Which is equivalent to

$$F \circ A(x) = B \circ F(x).$$

□

Again, we can easily observe that $\mathrm{Aut}_{\mathrm{LE}}(F)$ satisfies the conditions of a finite group, since it is sufficient to show that for $r, s \in \mathrm{Aut}_{\mathrm{LE}}(F)$ it is true that $rs \in \mathrm{Aut}_{\mathrm{LE}}(F)$. Since $r, s \in \mathrm{Aut}_{\mathrm{LE}}(F)$, therefore r and s are of the form

$$r = \begin{pmatrix} A_r & 0 \\ 0 & B_r \end{pmatrix}, \quad s = \begin{pmatrix} A_s & 0 \\ 0 & B_s \end{pmatrix}.$$

From the properties of matrix multiplication we have

$$rs = \begin{pmatrix} A_r & 0 \\ 0 & B_r \end{pmatrix} \begin{pmatrix} A_s & 0 \\ 0 & B_s \end{pmatrix} = \begin{pmatrix} A_r A_s & 0 \\ 0 & B_r B_s \end{pmatrix}.$$

Since we need to show that $rs \in \mathrm{Aut}_{\mathrm{LE}}(F)$, we need to show that the equation $F \circ A_r A_s = B_r B_s \circ F$ is satisfied.

$$F \circ A_r A_s = (B_r \circ F) \circ A_s = B_r \circ (F \circ A_s) = B_r \circ B_s \circ F$$

Thus we have shown that $rs \in \mathrm{Aut}_{\mathrm{LE}}(F)$, which implies that $\mathrm{Aut}_{\mathrm{LE}}(F)$ is a group. This group is finite, since we are in the finite field. For the purpose of the following theory we define similarity of matrices.

Definition 30. *Let M, M' be two matrices from $\mathrm{GL}(n, \mathbb{F}_2)$. We say that these two matrices are* similar*, denoted $M \sim M'$, if there exists a matrix $P \in \mathrm{GL}(n, \mathbb{F}_2)$ such that $M' = P^{-1}MP$.*

Note that similarity is an equivalence relation. Reflexivity is given by $M = I_n M I_n$. Symmetry is given by $M' = P^{-1}MP \implies M = PMP^{-1} \implies M' \sim M$. For the remaining transitivity let us assume that $M' \sim M$ and $N \sim M'$. Therefore exists $Q \in \mathrm{GL}(n, \mathbb{F}_2)$ such that $N = Q^{-1}M'Q$. Thus $N = Q^{-1}P^{-1}MPQ \implies N \sim M$.

We want to study vectorial Boolean function with non-trivial linear self-equivalence (meaning with non-trivial element in its subgroup $\mathrm{Aut}_{\mathrm{LE}}$) up to linear-equivalence. Let us suppose that two vectorial Boolean functions $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ are linearly equivalent. The natural question is what we can say about $\mathrm{Aut}_{\mathrm{LE}}(F)$ and $\mathrm{Aut}_{\mathrm{LE}}(G)$. We will show that vectorial Boolean functions F, G are linear-equivalent if and only if for $(A, B) \in \mathrm{Aut}_{\mathrm{LE}}(F)$ there exists (A', B'), where $A' \sim A$, $B' \sim B$, such that $(A', B') \in \mathrm{Aut}_{\mathrm{LE}}(G)$.

Suppose that $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a vectorial Boolean function. Let $(A, B) \in \mathrm{Aut}_{\mathrm{LE}}(F)$. Assume that $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a vectorial Boolean function, such that F and G are linear-equivalent. This means from the definition that there exist $L, L' \in \mathrm{GL}(n, \mathbb{F}_2)$ such that $G = L' \circ F \circ L$. We define $A' := L^{-1}AL$ and $B' := L'BL'^{-1}$, therefore we have $A = LA'L^{-1}$, which implies that $A' \sim A$, and $B = L'^{-1}B'L'$, which implies that $B' \sim B$. Hence

$$\begin{aligned}
F \circ A &= B \circ F \\
F \circ LA'L^{-1} &= L'^{-1}B'L' \circ F \\
L' \circ F \circ L \circ A' &= B' \circ L' \circ F \circ L \\
G \circ A' &= B' \circ G.
\end{aligned}$$

This means that for any vectorial Boolean function G which is linear-equivalent to F, we have some matrices A', B' which are similar to A, B such that $(A', B') \in \mathrm{Aut}_{\mathrm{LE}}(G)$.

Suppose again that $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a vectorial Boolean function. Let $(A, B) \in \mathrm{Aut}_{\mathrm{LE}}(F)$. Now let us assume that $A \sim A'$ and $B \sim B'$ for some $A', B' \in \mathrm{GL}(n, \mathbb{F}_2)$. This implies that there exist $P, Q \in \mathrm{GL}(n, \mathbb{F}_2)$ such that $A' = P^{-1}AP \implies A = PA'P^{-1}$ and $B' = Q^{-1}BQ \implies B = QB'Q^{-1}$. Thus

$$\begin{aligned}
F \circ A &= B \circ F \\
F \circ \left(PA'P^{-1}\right) & \left(QB'Q^{-1}\right) \circ F \\
\left(Q^{-1} \circ F \circ P\right) A' &= B' \circ \left(Q^{-1} \circ F \circ P\right).
\end{aligned}$$

We can define the function $G := Q^{-1} \circ F \circ P$. Therefore G is linear-equivalent to F and $(A', B') \in \mathrm{Aut}_{\mathrm{LE}}(G)$.

This gives us the notion that for a fixed vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ if we take any (A, B) from $\mathrm{Aut}_{\mathrm{LE}}(F)$ and we take any (A', B') such that $A \sim A'$ and $B \sim B'$, then we have a linear-equivalent function G for such $(A', B') \in \mathrm{Aut}_{\mathrm{LE}}(G)$.

3.2 Modules

The definition of $\mathrm{Aut}_{\mathrm{LE}}(F)$ consists of tuples of invertible matrices. For a matrix $A \in \mathrm{GL}(n, \mathbb{F}_2)$ we can define an equivalence class consisting of matrices similar to A. This implies that we can divide $\mathrm{GL}(n, \mathbb{F}_2)$ into the equivalence classes up to similarity. We are interested in a representative of these classes. To find such a representative we need to introduce the theory of modules. Every $\mathbb{F}_2[x]$-module $\mathbb{F}_2^n$ is defined with an action, which can be represented by a linear transformation $T : \mathbb{F}_2^n \to \mathbb{F}_2^n$. Every linear transformation T can be represented by an $n \times n$ matrix A, which means that for every matrix A we can define a linear transformation T which defines $\mathbb{F}_2[x]$-module $\mathbb{F}_2^n$. Using the Theorem 18 we will show that every $\mathbb{F}_2[x]$-module $\mathbb{F}_2^n$ can be decomposed into direct sum of modules, giving us a matrix in rational canonical form (RCF). This implies that for any matrix A we can find matrix in RCF. Furthermore, we will show that the matrix A is similar to a matrix in RCF, which implies that we can consider matrices in RCF as representatives of these similarity equivalence classes.

Let us begin with two theorems about modules. Both are from [4, Section 12.1, Theorems 4 and 5]. The first theorem (Theorem 17) gives us an insight into the basis of submodules of free modules. We will not prove it. The detailed proof can be found in the cited source.

The second theorem (Theorem 18) gives us a decomposition of the $\mathbb{F}_2[x]$-module $\mathbb{F}_2^n$ into a factor form. This theorem can be found in a more general form in the cited source, but since we are only interested in the $\mathbb{F}_2[x]$-module $\mathbb{F}_2^n$, we will state and prove the theorem specifically for this case and in a more detailed way. The statement and proof follows from [4, Section 12.1].

Theorem 17. *Let R be a P.I.D., let M be a free R-module of finite rank n and let N be a submodule of M. Then*

1. *N is free of rank m, $m \leq n$, $m \in \mathbb{N}$ and*
2. *there exists a basis $y_1, y_2, \dots, y_n$ of M so that $a_1y_1, a_2y_2, \dots, a_my_m$ is a basis of N where $a_1, a_2, \dots, a_m$ are nonzero elements of R with the divisibility relations*
$$a_1 \mid a_2 \mid \cdots \mid a_m.$$

Theorem 18. *Let $\mathbb{F}_2^n$ be a vector space of finite dimension $n \in \mathbb{N}$ over finite field $\mathbb{F}_2$. Consider $\mathbb{F}_2^n$ as a finitely generated $\mathbb{F}_2[x]$-module. Then*

$$\mathrm{Tor}(\mathbb{F}_2^n) \cong \mathbb{F}_2[x]/(a_1) \oplus \mathbb{F}_2[x]/(a_2) \oplus \cdots \oplus \mathbb{F}_2[x]/(a_m),$$

where $a_1, a_2, \dots, a_m$ are elements from $\mathbb{F}_2[x]$ which are not units in $\mathbb{F}_2[x]$ and which satisfy the divisibility relations

$$a_1 \mid a_2 \mid \cdots \mid a_m,$$

for some $m \in \mathbb{N}$ such that $m \leq n$.

Proof. The vector space $\mathbb{F}_2^n$ can be generated by a finite set of n elements. Let us denote them $g_1, \dots, g_n \in \mathbb{F}_2^n$. Now let us consider $\mathbb{F}_2[x]$-module $(\mathbb{F}_2[x])^n$. By Proposition 4 we know that this module is a free $\mathbb{F}_2[x]$-module of rank n with some basis which we denote $b_1, \dots, b_n$. Now we define a homomorphism π in such a way that for all $i \in \{1, \dots, n\}$ we have

$$\begin{aligned} \pi : (\mathbb{F}_2[x])^n &\to \mathbb{F}_2^n \\ b_i &\mapsto g_i. \end{aligned}$$

Such a mapping is surjective, since all g_i are mapped to by b_i and elements g_i generates $\mathbb{F}_2^n$. Now we can apply the first isomorphism theorem (Theorem 1). We know that $\mathrm{Ker}(\pi)$ is a submodule (Subsection 1.2.1) of $(\mathbb{F}_2[x])^n$, thus

$$(\mathbb{F}_2[x])^n / \mathrm{Ker}(\pi) \cong \mathbb{F}_2^n \,.$$

This gives us conditions that satisfy the conditions of Theorem 17. In particular, we have $\mathbb{F}_2[x]$ which is P.I.D., $(\mathbb{F}_2[x])^n$ is a free module of rank n over $\mathbb{F}_2[x]$ and $\mathrm{Ker}(\pi)$ which is a submodule of $(\mathbb{F}_2[x])^n$. Thus by Theorem 17 we have a basis $y_1, \dots, y_n$ of $(\mathbb{F}_2[x])^n$ so that $a_1y_1, \dots, a_ky_k$ is a basis of $\mathrm{Ker}(\pi)$, for some $k \in \mathbb{N}$, $k \leq n$, where $a_1, a_2, \dots, a_k$ are nonzero elements of $\mathbb{F}_2[x]$ with the divisibility relations $a_1 \mid a_2 \mid \cdots \mid a_k$.

Since we have the basis $y_1, \dots, y_n$ of the $\mathbb{F}_2[x]$-module $(\mathbb{F}_2[x])^n$, we know that for any nonzero element α from $(\mathbb{F}_2[x])^n$ there exist unique nonzero $r_1, \dots, r_n$ such that $\alpha = r_1y_1 + \cdots + r_ny_n$. The i-th summand r_iy_i is from the cyclic submodule $\mathbb{F}_2[x]y_1$. We can apply Proposition 2 and we get that

$$(\mathbb{F}_2[x])^n \cong \mathbb{F}_2[x]y_1 \oplus \mathbb{F}_2[x]y_2 \oplus \cdots \oplus \mathbb{F}_2[x]y_n.$$

The same can be done with the submodule $\mathrm{Ker}(\pi)$. Therefore

$$\mathrm{Ker}(\pi) \cong \mathbb{F}_2[x]a_1y_1 \oplus \mathbb{F}_2[x]a_2y_2 \oplus \cdots \oplus \mathbb{F}_2[x]a_ky_k.$$

This gives us

$$\mathbb{F}_2^n \cong (\mathbb{F}_2[x]y_1 \oplus \cdots \oplus \mathbb{F}_2[x]y_n)/(\mathbb{F}_2[x]a_1y_1 \oplus \cdots \oplus \mathbb{F}_2[x]a_ky_k).$$

We want to examine the right-hand side. For this let us define the mapping φ which is an $\mathbb{F}_2[x]$-homomorphism

$$\varphi : \mathbb{F}_2[x]y_1 \oplus \cdots \oplus \mathbb{F}_2[x]y_n \to \mathbb{F}_2[x]/(a_1) \oplus \cdots \oplus \mathbb{F}_2[x]/(a_k) \oplus (\mathbb{F}_2[x])^{n-k},$$

which maps elements as follows

$$(\alpha_1y_1, \dots, \alpha_ny_n) \mapsto (\alpha_1 \mod (a_1), \dots, \alpha_k \mod (a_k), \alpha_{k+1}, \dots, \alpha_n).$$

Now we are interested in $\mathrm{Ker}(\varphi)$. It is clear that for $i \in \{1, \dots, k\}$ the map φ maps the i-th component to zero if and only if $\alpha_i \mod (a_i) = 0 \iff \alpha_i = \beta_ia_i$ for some $\beta_i \in \mathbb{F}_2[x] \iff \alpha_iy_i \in \mathbb{F}_2[x]a_iy_i$.

For $i \in \{k+1, \dots, n\}$ the map φ maps the i-th component to zero if and only if $\alpha_i = 0$. Therefore

$$\mathrm{Ker}(\varphi) = (\mathbb{F}_2[x]a_1y_1 \oplus \mathbb{F}_2[x]a_2y_2 \oplus \cdots \oplus \mathbb{F}_2[x]a_ky_k)$$

and we can use the first isomorphism theorem again, so

$$(\mathbb{F}_2[x]y_1 \oplus \cdots \oplus \mathbb{F}_2[x]y_n)/\mathrm{Ker}(\varphi) \cong \mathbb{F}_2[x]/(a_1) \oplus \cdots \oplus \mathbb{F}_2[x]/(a_k) \oplus (\mathbb{F}_2[x])^{n-k}.$$

If we now combine this isomorphism with the one for map π, we get that

$$\mathbb{F}_2^n \cong \mathbb{F}_2[x]/(a_1) \oplus \cdots \oplus \mathbb{F}_2[x]/(a_k) \oplus (\mathbb{F}_2[x])^{n-k}.$$

Now we have to show that the module $(\mathbb{F}_2[x])^{n-k}$ does not appear in the direct sum, given that $\mathrm{Tor}(\mathbb{F}_2^n) = \mathbb{F}_2^n$. For the sake of contradiction, let us assume that $n-k > 0$. Let us take

$$p \in \mathbb{F}_2[x]y_1 + \mathbb{F}_2[x]y_2 + \cdots + \mathbb{F}_2[x]y_k + \mathbb{F}_2[x]y_{k+1} + \cdots + \mathbb{F}_2[x]y_n,$$

such that

$$p = a_1y_1 + a_2y_2 + \cdots + a_ky_k + p_{k+1}y_{k+1} + \cdots + p_ny_n$$

for some nonzero $p_{k+1}, \ldots, p_n \in \mathbb{F}_2[x]$. From the Subsection 1.2.2 we know that $\mathbb{F}_2^n = \mathrm{Tor}(\mathbb{F}_2^n)$, so there exist nonzero $\beta \in \mathbb{F}_2[x]$ such that $\beta p = 0$. Therefore we have

$$\beta p = \beta a_1y_1 + \beta a_2y_2 + \cdots + \beta a_ky_k + \beta p_{k+1}y_{k+1} + \cdots + \beta p_ny_n.$$

If we apply the map φ on βp we get

$$\varphi(\beta p) = (0, \ldots, 0, \beta p_{k+1}, \ldots, \beta p_n),$$

where $\beta p_{k+1}, \ldots, \beta p_n$ are nonzero elements. But since $\beta p = 0$, we have the following equation

$$\underbrace{(0, \ldots, 0)}_{n \text{ times}} = \varphi(0) = \varphi(\beta p) = (0, \ldots, 0, \beta p_{k+1}, \ldots, \beta p_n),$$

which contradicts φ being a $\mathbb{F}_2[x]$-homomorphism, since βp_i for $i \in \{k+1, \ldots, n\}$ is nonzero. This implies that the submodule $(\mathbb{F}_2[x])^{n-k}$ cannot be in the direct sum, thus we have

$$\mathrm{Tor}(\mathbb{F}_2^n) = \mathbb{F}_2^n \cong \mathbb{F}_2[x]/(a_1) \oplus \cdots \oplus \mathbb{F}_2[x]/(a_k).$$

Finally, if $a_i = 1$, then $\mathbb{F}_2[x]/(a_i) = 0$, since the ideal $(a_i) = \mathbb{F}_2[x]$. This implies that we can remove from the direct sum all $\mathbb{F}_2[x]/(a_i)$ for which $a_i = 1$. Let us denote the number of such submodules by t, so we define $m := k - t$. Hence

$$\mathrm{Tor}(\mathbb{F}_2^n) = \mathbb{F}_2^n \cong \mathbb{F}_2[x]/(a_1) \oplus \cdots \oplus \mathbb{F}_2[x]/(a_m).$$

Note also that the the factor form $a_1, \ldots, a_m$ is unique. Let $b_1, \ldots, b_{m'}$ be another factor form, where $m' \in \mathbb{N}$, such that

$$\begin{aligned}\mathbb{F}_2^n &\cong \mathbb{F}_2[x]/(a_1) \oplus \cdots \oplus \mathbb{F}_2[x]/(a_m)\\ &\cong \mathbb{F}_2[x]/(b_1) \oplus \cdots \oplus \mathbb{F}_2[x]/(b_{m'}).\end{aligned}$$

For a_i, where $i \in \{1, \ldots, m\}$ we know that $a_1y_1, \ldots, a_my_m$ is a basis of a free submodule $\mathrm{Ker}(\pi)$. Let $y'_1, \ldots, y'_{m'}$ be such basis of $\mathbb{F}_2^n$ that $b_1y'_1, \ldots, b_{m'}y'_{m'}$ is another basis of a free submodule $\mathrm{Ker}(\pi)$. Since the submodule is free, it follows that any element from $\mathrm{Ker}(\pi)$ can be express in a unique way. Thus let $v \in \mathrm{Ker}(\pi)$, thus

$$v = v_1a_1y_1 + \cdots + v_ma_my_m = v'_1b_1y'_1 + \cdots + v'_{m'}b_{m'}y'_{m'},$$

which implies that $m = m'$ and since a_i and b_j are ordered, we know that $(a_i) = (b_i)$ for $i \in \{1, \dots, m\}$, thus $a_i = b_i$.

The isomorphism implies that $m = m'$ and that $(a_i) = (b_i)$, which implies that $a_i = b_i$. Therefore we can define invariant factors of $\mathbb{F}_2^n$.

$\square$

Note that the sum of the degrees of the polynomials $a_1, \dots, a_m$ is equal to n. This follows from the fact that the cardinality of the left and right sides must be equal due to isomorphism. The cardinality of $\mathbb{F}_2^n$ is 2^n and the cardinality of $\mathbb{F}_2[x]/(a_i)$ is equal to $2^{\deg a_i}$. Therefore $2^n = 2^{\deg a_1} 2^{\deg a_2} \cdots 2^{\deg a_m} \implies n = \deg a_1 + \deg a_2 + \cdots + \deg a_m$.

Definition 31. *Let $a_1, \dots, a_m \in \mathbb{F}_2[x]$ be elements from Theorem 18. Then we call them* invariant factors of $\mathbb{F}_2^n$.

So far we know that for a fixed choice of transformation T the elements $a_1, \dots, a_m \in \mathbb{F}_2[x]$ from Theorem 18 are invariants. The next theorem gives us that the invariant factors are invariants up to a module isomorphism, i.e. if two R-modules M_1 and M_2, each defined with different action, are isomorphic, then it implies that the invariant factor form is the same for both of these transformations.

With the definition of invariant factors we can state the following theorem from [4, Section 12.1, Theorem 9]. The detailed proof can be found in the cited source.

Theorem 19. *Let R be P.I.D. Two finitely generated R-modules M_1 and M_2 are isomorphic if and only if they have the same free rank and the same list of invariant factors.*

Now we want to examine what it means to use the element $\beta = x \in \mathbb{F}_2[x]$ on the left (and therefore on the right side) of the isomorphism $\mathbb{F}_2^n \cong \mathbb{F}_2[x]/(a_1) \oplus \cdots \oplus \mathbb{F}_2[x]/(a_m)$. For this we will use some parts of [4, Section 12.2].

First we see that this operation is valid, since we consider $\mathbb{F}_2^n$ to be a $\mathbb{F}_2[x]$-module, and since $\beta \in \mathbb{F}_2[x]$ we know that the image of β via such an isomorphism is still an element β. Suppose we have some $v = (v_1, \dots, v_n) \in \mathbb{F}_2^n$ which is isomorphic to $(\alpha_1 \mod (a_1(x)), \dots, \alpha_m \mod (a_m(x)))$, where $\alpha_i \in \mathbb{F}_2[x]$ and a_i are from the Theorem 18. Thus, if we use β on the left side, we get the action of x on the $\mathbb{F}_2^n$, which we examined in the Subsection 1.2.2 and thus

$$\beta v = Tv,$$

where T is any linear transformation. If we apply the same element β to the right side of the isomorphism, we see that

$$\begin{aligned}
&\beta(\alpha_1 \mod (a_1(x)), \dots, \alpha_m \mod (a_m(x))) \\
&= (\beta\alpha_1 \mod (a_1(x)), \dots, \beta\alpha_m \mod (a_m(x))) \\
&= (x\alpha_1 \mod (a_1(x)), \dots, x\alpha_m \mod (a_m(x))).
\end{aligned}$$

This gives us a feel for the behaviour of any linear transformation T. We see that the transformation T behaves like a multiplication with the polynomial x on the right side of the isomorphism.

Since the element β on the right side is used component-wise, we will examine the first component $\mathbb{F}_2[x]/(a_1)$ without loss of generality. Let us assume that $a_1(x) = x^k + b_{k-1}x^{k-1} + \cdots + b_1x + b_0 \in \mathbb{F}_2[x]$, where $k \in \mathbb{N}$. Therefore in $\mathbb{F}_2[x]/(a_1)$ we have that $a_1(x) = 0 \implies x^k = b_0 + b_1x + \cdots + b_{k-1}x^{k-1}$.

We can consider the $\mathbb{F}_2[x]/(a_1)$ as a finite vector space over the finite field $\mathbb{F}_2$ with the basis $1, x, x^2, \ldots, x^{k-1}$. This gives us the option to consider the action of $\beta = x$ with respect to this basis. So the x acts as follows: $1 \mapsto x$, $x \mapsto x^2$, ..., $x^{k-2} \mapsto x^{k-1}$, $x^{k-1} \mapsto x^k = b_0 + b_1x + \cdots + b_{k-1}x^{k-1}$.

The multiplication by x can be represented by a $k \times k$ matrix with respect to the basis. Such a matrix is called a companion matrix and is subject to the following definition.

Definition 32. *Let $p(x) = x^k + b_{k-1}x^{k-1} + \cdots + b_1x + b_0$ be any monic polynomial in $\mathbb{F}_2[x]$. The* companion matrix *of $p(x)$ is the $k \times k$ matrix with 1's down the first subdiagonal, $-b_0, -b_1, \ldots, -b_{k-1}$ down the last column and zeros elsewhere. The companion matrix of $p(x)$ will be denoted $\mathcal{C}_{p(x)}$.*

$$\mathcal{C}_{p(x)} = \begin{pmatrix} 0 & 0 & \cdots & \cdots & \cdots & -b_0 \\ 1 & 0 & \cdots & \cdots & \cdots & -b_1 \\ 0 & 1 & \cdots & \cdots & \cdots & -b_2 \\ 0 & 0 & \ddots & & & \vdots \\ \vdots & \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & \cdots & 1 & -b_{k-1} \end{pmatrix}$$

This can be applied to any of the modules in the direct sum on the right side of the isomorphism. Since $\mathbb{F}_2[x]/(a_i)$ is a free module for $i \in \{1, \ldots, m\}$, we can choose the basis of such a submodule. For this basis we can find the corresponding elements in $\mathbb{F}_2^n$. Let us denote the set of such elements as β_i. This means that an arbitrary linear transformation T acts on β_i via the companion matrix $\mathcal{C}_{a_i(x)}$.

Now we can take the union of β_i and denote it as $\beta = \bigcup_{i=1}^{m} \beta_i$. Then β forms the basis of $\mathbb{F}_2^n$, because we know from the Proposition 3 that the submodules in the direct sum are disjoint. Thus, if we define the matrix $\mathcal{C} := \operatorname{diag}(\mathcal{C}_{a_1(x)}, \ldots, \mathcal{C}_{a_m(x)})$, then we know that, with the respect to the basis β, we can represent the action of a linear transformation T on the left side of the isomorphism by the matrix $\mathcal{C}$ on the right side of the isomorphism.

From the Theorems 18 and 19, we know that the matrix is uniquely determined by the factor form up to the isomorphism of $\mathbb{F}_2^n$ as a $\mathbb{F}_2[x]$-module. Knowing the uniqueness of the factors, we can define the following. The definition is from [4, Section 12.2]

Definition 33. *We say that a matrix is in a* rational canonical form *if it is of the form* $\operatorname{diag}(\mathcal{C}_{a_1(x)}, \ldots, \mathcal{C}_{a_m(x)})$ *for monic polynomials $a_1(x), \ldots, a_m(x)$ of degree at least one with the condition*

$$a_1(x) \mid a_2(x) \mid \cdots \mid a_m(x).$$

A rational canonical form *for a linear transformation T is a matrix representing T which is in rational canonical form.*

With this definition we can say that every linear transformation T has a rational canonical form. We are interested in whether this rational canonical form is unique. For this we will use the following theorem which is based on the theorem from [4, Section 12.2] for which we will give a proof that follows the discussion from the same source.

Theorem 20. *Let $\mathbb{F}_2^n$ be a finite dimensional vector space over the field $\mathbb{F}_2$ and let T be a linear transformation of $\mathbb{F}_2^n$.*

There is a basis for $\mathbb{F}_2^n$ with respect to which the matrix for T is in rational canonical form, i.e., is a block diagonal matrix whose diagonal blocks are the companion matrices for monic polynomials $a_1(x), a_2(x), \ldots, a_m(x)$ of degree at least one with $a_1(x) \mid a_2(x) \mid \cdots \mid a_m(x)$.

The rational canonical form for T is unique.

Proof. The existence of such a basis was proved in the proof of Theorem 18. The rational canonical form of the matrix for T was discussed after the proof.

We need need to prove the uniqueness of the rational canonical form for T. Suppose that $b_1(x), b_2(x), \ldots, b_t(x)$ are monic polynomials in $\mathbb{F}_2[x]$ of degree at least one such that $b_1(x) \mid b_2(x) \mid \cdots \mid b_t(x)$ for some $t \in \mathbb{N}$. Suppose also that $\mathcal{E}$ is a basis of $\mathbb{F}_2^n$ for which the matrix of T is of the form $\operatorname{diag}(\mathcal{C}_{b_1(x)}, \ldots, \mathcal{C}_{b_t(x)})$ where $\mathcal{C}_{b_i(x)}$ for $i \in \{1, \ldots, t\}$ is a companion matrix.

Let D_i be the i-th subspace of $\mathbb{F}_2^n$ such that the matrix of T on D_i is the $\mathcal{C}_{b_i(x)}$. The vector space $\mathbb{F}_2^n$ is therefore isomorphic to the direct sum of the subspaces D_i. Since $\mathbb{F}_2[x]$ is P.I.D. and we now consider $\mathbb{F}_2^n$ to be a $\mathbb{F}_2[x]$-module, we know that there must be an element which generates each D_i. Therefore, D_i is a cyclic $\mathbb{F}_2[x]$-module.

If we now apply the Theorem 18 on the $\mathbb{F}_2^n$, we get some $a_1(x) \mid \cdots \mid a_m(x)$ with the properties according to the theorem. Now we can use the Theorem 19, so we have that $a_i(x)$ and $b_i(x)$ must differ by a unit factor in $\mathbb{F}_2[x]$, but since we choose $b_i(x)$ as monic polynomials, therefore we have $a_i(x) = b_i(x)$ for $i \in \{1, \ldots, t = m\}$.

□

So far, we have presented the theory in terms of linear transformations. For the following part we will switch to the notion of matrices. Consider the $n \times n$ matrix A over the finite field $\mathbb{F}_2$. We can define a linear transformation $T : \mathbb{F}_2^n \to \mathbb{F}_2^n$ as $T(v) = Av$ for any $v \in \mathbb{F}_2^n$. Therefore, any statement that uses a linear transformation can be expressed using a matrix instead. The following theorem is given without proof as Lemma 3 in [3, Section 4].

Theorem 21. *Every element $M \in \mathrm{GL}(n, \mathbb{F}_2)$ is similar to a unique $M' \in \mathrm{GL}(n, \mathbb{F}_2)$ of the form $M' = \operatorname{diag}(\mathcal{C}_{a_m(x)}, \mathcal{C}_{a_{m-1}(x)}, \ldots, \mathcal{C}_{a_1(x)})$ for polynomials a_i such that $a_m \mid a_{m-1} \mid \cdots \mid q_1$. This matrix M' is called the rational canonical form of M, denoted $\mathrm{RCF}(M)$.*

Proof. The existence of such a matrix M' is given by the Theorem 20. Since we assume that $M \in \mathrm{GL}(n, \mathbb{F}_2)$, we know that such a matrix represents an isomorphic linear transformation. So, M' is also a linear transformation, which is bijective, thus $M' \in \mathrm{GL}(n, \mathbb{F}_2)$.

For similarity we know that M' can be obtained from M by choosing the right basis of $\mathbb{F}_2^n$. This means that there exists a matrix $U \in \mathrm{GL}(n, \mathbb{F}_2)$ which represents the transition from one basis to the basis for which the matrix M is in RCF. So $M = U^{-1}\mathcal{C}U$, which is the definition of similarity.

□

3.3 Representatives up to Power-Similarity

In this section we start with a proposition that gives us three possible cases of orders of a tuple from the automorphism group $\mathrm{Aut}_{\mathrm{LE}}$. For one of these cases we can introduce the equivalence relation power-similarity, which extends the definition of similarity. We will prove in Theorem 24 that the power-similarity implies linear-equivalence. Therefore, for the classification of functions $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, with a non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$, up to linear-equivalence, we can only consider tuples $(\mathrm{RCF}(A), \mathrm{RCF}(B))$ up to power-similarity.

The following proposition is based on the lemmas in [3, Section 4]. The proof is described in more detail in this thesis.

Proposition 22. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ for which there exists a non-trivial automorphism in $\mathrm{Aut}_{\mathrm{LE}}(F)$. Then there exist $A, B \in \mathrm{GL}(n, \mathbb{F}_2)$ with $F \circ A = B \circ F$ such that either*

1. $\mathrm{ord}(A) = \mathrm{ord}(B) = p$ *for p prime, or*
2. $A = I_n$ *and* $\mathrm{ord}(B) = p$ *for p prime, or*
3. $B = I_n$ *and* $\mathrm{ord}(A) = p$ *for p prime.*

Proof. Suppose $g \in \mathrm{Aut}_{\mathrm{LE}}(F)$ is the non-trivial automorphism (thus $g \neq I_{2n}$). This means from the definition of $\mathrm{Aut}_{\mathrm{LE}}(F)$ and Proposition 16 that

$$g \in \{\mathrm{diag}(A, B) \in \mathrm{GL}(2n, \mathbb{F}_2) \mid A, B \in \mathrm{GL}(n, \mathbb{F}_2) \text{ and } F \circ A = B \circ F\}.$$

Since g is a non-trivial element of the group $\mathrm{Aut}_{\mathrm{LE}}(F)$, hits implies that there exists an element $h \in \langle g \rangle$ such that the order of the subgroup, generated by this element, is prime. Denote this as $p := \mathrm{ord}(h)$.

Let us now assume, that $h = \mathrm{diag}(A, B)$ for some fixed A, B, where $A \neq I_n \neq B$. This implies, that the order the of matrix h is equal to the least common multiple of $\mathrm{ord}(A)$ and $\mathrm{ord}(B)$. Thus $\mathrm{ord}(h) = x \cdot \mathrm{ord}(A) = y \cdot \mathrm{ord}(B)$ for some $x, y \in \mathbb{N}$. As we mentioned before, the order of h is prime number, thus $x = y = 1$, which implies, that $p = \mathrm{ord}(A) = \mathrm{ord}(B)$.

Now let us assume, that $A = I_n$ and $B \neq I_n$. This implies that the order of A is equal to one and the order of B is equal to some prime number p. Just like in the previous case, the order of h is equal to the least common multiple of these two orders, therefore the order of B is equal to p. The last case where $A \neq I_n$ and $B = I_n$ is similar to the previous one.

□

The following definition is from [3, Section 4]. It is given only for matrices $A, B, C, D \in \mathrm{GL}(n, \mathbb{F}_2)$ of order p for p prime. This implies that this definition can only be used at point 1 of the Proposition 22. We will prove in Proposition 23 that the power-similarity is an equivalence relation.

Definition 34. *Let $A, B, C, D \in \mathrm{GL}(n, \mathbb{F}_2)$ be of order p for p prime. The tuple (A, B) is said to be* power-similar *to the tuple (C, D), denoted $(A, B) \sim_p (C, D)$ if there exists a positive integer i such that $A \sim C^i$ and $B \sim D^i$.*

Proposition 23. *Relation being power-similar is a relation of equivalence.*

Proof. The equivalence relation must satisfy reflexivity, symmetry and transitivity. Let us denote $p \in \mathbb{N}$ be prime.

For reflexivity, let us assume that $(A, B) \in \mathrm{GL}(n, \mathbb{F}_2)$, where A, B are of the order p. We need to show that $(A, B) \sim_p (A, B)$. This is trivial since $A = I_n A I_n$ and $B = I_n B I_n$.

For symmetry, let us assume that $(A, B) \in \mathrm{GL}(n, \mathbb{F}_2)$ and $(C, D) \in \mathrm{GL}(n, \mathbb{F}_2)$, where A, B, C, D are of the order p. We need to show that $(A, B) \sim_p (C, D) \iff (C, D) \sim_p (A, B)$.

For the first implication. We assume that $(A, B) \sim_p (C, D)$, then $A \sim C^i$ and $B \sim D^i$, therefore $C^i = P^{-1}AP$ and $D^i = Q^{-1}BQ$ for some $P, Q \in \mathrm{GL}(n, \mathbb{F}_2)$. Thus $A = PC^iP^{-1}$ and $B = QD^iQ^{-1}$. This means that $C^i \sim A$ and $D^i \sim B$, which is equivalent to $(C, D) \sim_p (A, B)$. The second implication can be proved in the same way.

For transitivity, suppose that for $A, B, C, D, E, F \in \mathrm{GL}(n, \mathbb{F}_2)$ (all of the order p), we have that $(A, B) \sim_p (C, D)$ and $(C, D) \sim_p (E, F)$. We want to show that this implies $(A, B) \sim_p (E, F)$. Since $(A, B) \sim_p (C, D)$, then we know from symmetry that $(C, D) \sim_p (A, B)$, thus we have matrices $P, Q \in \mathrm{GL}(n, \mathbb{F}_2)$ such that $A^i = P^{-1}CP$ and $B^i = Q^{-1}DQ$, which implies that $C = PA^iP^{-1}$ and $D = QB^iQ^{-1}$. Also, since $(C, D) \sim_p (E, F)$, then we have matrices $R, S \in \mathrm{GL}(n, \mathbb{F}_2)$ such that $E^j = R^{-1}CR$ and $F^j = S^{-1}DS$.

So if we replace C, D we get

$$E^j = R^{-1}CR = R^{-1}\left(PA^iP^{-1}\right)R = \left(R^{-1}P\right)A^i\left(P^{-1}R\right)$$
$$F^j = S^{-1}DS = S^{-1}\left(QB^iQ^{-1}\right) = \left(S^{-1}Q\right)B^i\left(Q^{-1}S\right)$$

From the Proposition 22 we know that for the maximum of $\{\mathrm{ord}(A), \mathrm{ord}(B)\}$, denoted as p, that $A^p = B^p = I_n$. Therefore for $k := i^{-1} \mod p$ we have that

$$\begin{aligned}
E^{jk} &= \left(R^{-1}CR\right)^k \\
&= \underbrace{\left(R^{-1}CR\right)\left(R^{-1}CR\right)\cdots\left(R^{-1}CR\right)}_{k \text{ times}} \\
&= R^{-1}C^kR \\
&= R^{-1}\left(PA^iP^{-1}\right)^kR \\
&= R^{-1}PA^{ik}P^{-1}R \\
&= \left(R^{-1}P\right)A\left(P^{-1}R\right).
\end{aligned}$$

This is equivalent to $A \sim E^{jk}$. We can do the same for F^{jk} which gives us $B \sim F^{jk}$. Therefore $(A, B) \sim_p (E, F)$.

$\square$

Now we can formulate the following theorem. It states that for every equivalence class of power-similarity we can find a tuple which is in rational canonical form. The following theorem and its proof are based on Lemma 4 from [3, Section 4]. The proof is described in more detail in this thesis.

Theorem 24. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ with an automorphism* $\operatorname{diag}(A, B) \in \operatorname{Aut}_{\mathrm{LE}}(F)$ *for $A, B \in \mathrm{GL}(n, \mathbb{F}_2)$ such that* $\operatorname{ord}(A) = \operatorname{ord}(B) = p$*, where p is prime. For every $(\widetilde{B}, \widetilde{A})$ power-similar to (B, A), there is a function G linear-equivalent to F such that* $\operatorname{diag}(\mathrm{RCF}(\widetilde{A}), \mathrm{RCF}(\widetilde{B})) \in \operatorname{Aut}_{\mathrm{LE}}(G)$.

Proof. Let us take an arbitrary tuple $(\widetilde{B}, \widetilde{A})$ which is power-similar to the tuple (B, A). This implies from the Definition 34 that there exists a positive integer i such that $B \sim \widetilde{B}^i$ and $A \sim \widetilde{A}^i$, which implies from the Definition 30 that there exist some matrices $P, Q \in \mathrm{GL}(n, \mathbb{F}_2)$ such that

$$A = P^{-1}\widetilde{A}^i P \quad \text{and} \quad B = Q^{-1}\widetilde{B}^i Q.$$

First we noticed that for any $k \in \mathbb{N}$ we have

$$\begin{aligned} A^{pk} &= \left(P^{-1}\widetilde{A}^i P\right)^{pk} \\ I_n &= P^{-1}\widetilde{A}^{ipk} P \\ PP^{-1} &= PP^{-1}\widetilde{A}^{ipk} PP^{-1} \\ I_n &= \widetilde{A}^{ipk} \end{aligned}$$

and if we choose $k := i^{-1} \mod p$, then we get $I_n = \widetilde{A}^p$. We can do the same with the B and therefore get $I_n = \widetilde{B}^p$.

Second we noticed that again for any $k \in \mathbb{N}$ we have that $A^k \sim \widetilde{A}^{ik}$ because

$$\begin{aligned} A^k &= \left(P^{-1}\widetilde{A}^i P\right)^k \\ &= \underbrace{(P^{-1}\widetilde{A}^i P)(P^{-1}\widetilde{A}^i P) \cdots (P^{-1}\widetilde{A}^i P)}_{k \text{ times}} \\ &= P^{-1}\widetilde{A}^{ik} P, \end{aligned}$$

so we also have that $B^k = Q^{-1}\widetilde{B}^{ik} Q$.

The last thing we noticed is that $F \circ A^k = B^k \circ F$ for any $k \in \mathbb{N}$. We know that $\operatorname{diag}(A, B) \in \operatorname{Aut}_{\mathrm{LE}}(F)$ implies that A, B satisfy $F \circ A = B \circ F$. It is trivial that for $k = 1$ the equation $F \circ A^k = B^k \circ F$ holds. Let us assume for induction that the equation holds for k, thus we want to show that it holds for $k + 1$.

$$F \circ A^{k+1} = (F \circ A^k) \circ A = (B^k \circ F) \circ A = B^k \circ (F \circ A) = B^k \circ B \circ F = B^{k+1} \circ F.$$

Now we can combine all these remarks together. Since we have $F \circ A^k = B^k \circ F$, we can replace A and B, therefore we get

$$F \circ \left(P^{-1}\widetilde{A}^{ik} P\right) = \left(Q^{-1}\widetilde{B}^{ik} Q\right) \circ F.$$

We apply matrix Q from the left and the matrix P^{-1} on both sides of the equation. This gives us

$$Q \circ F \circ P^{-1}\widetilde{A}^{ik} = \widetilde{B}^{ik} Q \circ F \circ P^{-1}.$$

We denote $G := Q \circ F \circ P^{-1}$ (this satisfies the definition of F, which is linearly-equivalent to G), so we have

$$G \circ \widetilde{A}^{ik} = \widetilde{B}^{ik} \circ G.$$

Since the k was arbitrary, we can choose $k = i^{-1} \mod p$. This gives us

$$G \circ \widetilde{A} = \widetilde{B} \circ G.$$

The relation of being power-similar forms equivalence, classes as we showed in Proposition 23. Therefore, without loss of generality, we can choose the matrices $\widetilde{A}, \widetilde{B}$ up to similarity. We have shown in Theorem 21 that for $\widetilde{A}, \widetilde{B}$ there exist unique matrices $\mathrm{RCF}(\widetilde{A}), \mathrm{RCF}(\widetilde{B})$. Thus we can choose $\widetilde{A}$ and $\widetilde{B}$ in their rational canonical form.

□

3.4 Algorithm

We can now use the statements from the previous sections of this chapter to search for vectorial Boolean functions up to LE-automorphism. Suppose that we have a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. From the Proposition 22 we know that for every $\mathrm{Aut}_{\mathrm{LE}}(F)$ we have at least one element which satisfies the proposition. Such elements are similar (or power-similar) to tuples of matrices in RCF.

From the Proposition 23 we know that we can establish equivalence classes up to power similarity on the elements satisfying the first point from Proposition 22. On the elements satisfying points 2 and 3 we can establish equivalence classes up to similarity. We can use tuples of matrices in RCF as representatives of each of the equivalence classes. So we find all RCF matrices of prime order and from these matrices we create tuples (RCF matrix, I_n) and (I_n, RCF matrix). Then we create tuples as all combinations of RCF matrices that are different from I_n and which have the same prime order. We check these tuples to see if they are pairwise power-similar, and if so, we remove such tuples, leaving only one tuple as a representative. This gives us all the representatives of the equivalence classes.

Such an equivalence class contains matrices that are similar (or power-similar) to matrices in the group $\mathrm{Aut}_{\mathrm{LE}}$ for functions that are linear-equivalent. Therefore, we have sorted vectorial Boolean functions with non-trivial element in their automorphism group up to linear-equivalence.

This implies that we can run an algorithm which, for a given dimension n, searches for all matrices in RCF of prime orders and from these matrices we can generate all representatives of equivalence classes. The implementation of the algorithm is attached in A.9.

Algorithm 3 Search for linear-equivalence classes

Output: list of linear-equivalence classes with representatives for each of them

```
1: $n \leftarrow$ integer representing the dimension in which we want to search for APN function
2: “RCF_matrices” $\leftarrow$ generate all possible RCF matrices
3: “all_prime_tuples” $\leftarrow$ generate all possible tuples of RCF matrices according to the point 1 in Proposition 22
4: “tuples_up_to_power-similarity” $\leftarrow$ test all of tuples from “all_prime_tuples” for power-similarity
5: “all_le_classes” $\leftarrow$ append classes according to the point 1 in Proposition 22 from “all_prime_tuples” and classes according to the points 2 and 3 in Proposition 22 from “RCF_matrices”
```

3.4.1 Description

As a first step, the algorithm generates all RCF matrices of the given dimension n that are of the prime order. This is done by generating companion matrices of prime order for dimensions less than or equal to n and combining them according to the Theorem 21. Then the algorithm generates all possible tuples where both matrices in the tuple are not equal to the identity matrix and both matrices have the same prime order. Such tuples are checked for power-similarity. If there are some tuples that are power-similar, then we select one tuple as representative of the equivalence classes. Tuples of the form where one of the matrices is equal to identity do not need to be checked further, since we have only defined similarity for these tuples, and each matrix is similar to a unique matrix in RCF.

3.4.2 Results

We run the Algorithm 3 for $n \in \{3, \dots, 12\}$. The specific results are attached in A.10. We were able to achieve the same results for $n = 7, 8, 9$ and 10 as in [2] up to power-similarity. The results for the remaining n have not been given.

Let us assume that $p \in \mathbb{N}$ is prime.

- $n = 3$

  We have found 12 linear-equivalence classes of functions $F : \mathbb{F}_2^3 \to \mathbb{F}_2^3$ which have non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$. For $\mathrm{ord}(A) = \mathrm{ord}(B) = p$ we have found 4 equivalence classes. For $\mathrm{ord}(A) = p$ and $B = I_3$ we have found 4 equivalence classes. For $A = I_3$ and $\mathrm{ord}(B) = p$ we have found 4 equivalence classes. The prime orders of matrices in RCF are $2, 3, 7$.

- $n = 4$

  We have found 25 linear-equivalence classes of functions $F : \mathbb{F}_2^4 \to \mathbb{F}_2^4$ which have non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$. For $\mathrm{ord}(A) = \mathrm{ord}(B) = p$ we have found 11 equivalence classes. For $\mathrm{ord}(A) = p$ and $B = I_4$ we have found 7 equivalence classes. For $A = I_4$ and $\mathrm{ord}(B) = p$ we have found 7 equivalence classes. The prime orders of matrices in RCF are $2, 3, 5, 7$.

- $n = 5$

We have found 43 linear-equivalence classes of functions $F : \mathbb{F}_2^5 \to \mathbb{F}_2^5$ which have non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$. For $\mathrm{ord}(A) = \mathrm{ord}(B) = p$ we have found 17 equivalence classes. For $\mathrm{ord}(A) = p$ and $B = I_5$ we have found 13 equivalence classes. For $A = I_5$ and $\mathrm{ord}(B) = p$ we have found 13 equivalence classes. The prime orders of matrices in RCF are $2, 3, 5, 7, 31$.

- $n = 6$

  We have found 74 linear-equivalence classes of functions $F : \mathbb{F}_2^6 \to \mathbb{F}_2^6$ which have non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$. For $\mathrm{ord}(A) = \mathrm{ord}(B) = p$ we have found 38 equivalence classes. For $\mathrm{ord}(A) = p$ and $B = I_6$ we have found 18 equivalence classes. For $A = I_6$ and $\mathrm{ord}(B) = p$ we have found 18 equivalence classes. The prime orders of matrices in RCF are $2, 3, 5, 7, 31$.

- $n = 7$

  We have found 128 linear-equivalence classes of functions $F : \mathbb{F}_2^7 \to \mathbb{F}_2^7$ which have non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$. For $\mathrm{ord}(A) = \mathrm{ord}(B) = p$ we have found 56 equivalence classes. For $\mathrm{ord}(A) = p$ and $B = I_7$ we have found 36 equivalence classes. For $A = I_7$ and $\mathrm{ord}(B) = p$ we have found 36 equivalence classes. The prime orders of matrices in RCF are $2, 3, 5, 7, 31, 127$.

- $n = 8$

  We have found 157 linear-equivalence classes of functions $F : \mathbb{F}_2^8 \to \mathbb{F}_2^8$ which have non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$. For $\mathrm{ord}(A) = \mathrm{ord}(B) = p$ we have found 75 equivalence classes. For $\mathrm{ord}(A) = p$ and $B = I_8$ we have found 41 equivalence classes. For $A = I_8$ and $\mathrm{ord}(B) = p$ we have found 41 equivalence classes. The prime orders of matrices in RCF are $2, 3, 5, 7, 17, 31, 127$.

- $n = 9$

  We have found 217 linear-equivalence classes of functions $F : \mathbb{F}_2^9 \to \mathbb{F}_2^9$ which have non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$. For $\mathrm{ord}(A) = \mathrm{ord}(B) = p$ we have found 111 equivalence classes. For $\mathrm{ord}(A) = p$ and $B = I_9$ we have found 53 equivalence classes. For $A = I_9$ and $\mathrm{ord}(B) = p$ we have found 53 equivalence classes. The prime orders of matrices in RCF are $2, 3, 5, 7, 17, 31, 73, 127$.

- $n = 10$

  We have found 401 linear-equivalence classes of functions $F : \mathbb{F}_2^{10} \to \mathbb{F}_2^{10}$ which have non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$. For $\mathrm{ord}(A) = \mathrm{ord}(B) = p$ we have found 247 equivalence classes. For $\mathrm{ord}(A) = p$ and $B = I_{10}$ we have found 77 equivalence classes. For $A = I_{10}$ and $\mathrm{ord}(B) = p$ we have found 77 equivalence classes. The prime orders of matrices in RCF are $2, 3, 5, 7, 11, 17, 31, 73, 127$.

- $n = 11$

  We have found 431 linear-equivalence classes of functions $F : \mathbb{F}_2^{11} \to \mathbb{F}_2^{11}$ which have non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$. For $\mathrm{ord}(A) = \mathrm{ord}(B) = p$

we have found 257 equivalence classes. For $\mathrm{ord}(A) = p$ and $B = I_{11}$ we have found 87 equivalence classes. For $A = I_{11}$ and $\mathrm{ord}(B) = p$ we have found 87 equivalence classes. The prime orders of matrices in RCF are $2, 3, 5, 7, 11, 17, 23, 31, 73,\ 89, 127$.

- $n = 12$

  We have found 536 linear-equivalence classes of functions $F : \mathbb{F}_2^{12} \to \mathbb{F}_2^{12}$ which have non-trivial element in $\mathrm{Aut}_{\mathrm{LE}}(F)$. For $\mathrm{ord}(A) = \mathrm{ord}(B) = p$ we have found 344 equivalence classes. For $\mathrm{ord}(A) = p$ and $B = I_{12}$ we have found 96 equivalence classes. For $A = I_{12}$ and $\mathrm{ord}(B) = p$ we have found 96 equivalence classes. The prime orders of RCF matrices are $2, 3, 5, 7, 11, 13, 17, 31, 73,\ 89, 127$.

4. Function Trimming

This chapter focuses on the other approach to finding APN functions. The basic idea of this method is to restrict a function in dimension n to the dimension $n-1$. We will introduce the notion of trimming a function and show how we can restrict the search space up to EA-equivalence. The functions obtained do not have to be APN, so we need to check this property. The first section introduces the notations and definitions. In the second section we will show that the EA-equivalence can reduce the search space, and in the third section we will introduce the algorithm based on the theory presented in the first two sections. In the this chapter we will use the information provided by [5, Section 3].

Our contribution is to present the theory in a clearer way, to prove the statements in more detail, to present our own implementation of the algorithm for finding a trim spectrum of a given function, which was approached independently from the implementation provided on the GitHub repository by the authors of [5], to describe the algorithm in the context of the theory from this chapter, to introduce a function for computing function $G : \mathbb{F}_2^{n-1} \to \mathbb{F}_2^{n-1}$ based on the trim of a function F (where $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$) and to replicate the results presented in [5] and make them available on the GitHub repository.

4.1 Definitions and Notations

We begin by defining the extension and restriction of two vectorial Boolean functions based on [5, Section 1].

Definition 35. *Let $n, m \in \mathbb{N}$ where $m < n$ and let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $G : \mathbb{F}_2^m \to \mathbb{F}_2^m$ be two vecotrial Boolean functions. We say that F is an* extension *of G and that G is* restriction *of F, denoted by $G \prec F$, if there exists an affine injective mapping $\phi : \mathbb{F}_2^m \to \mathbb{F}_2^n$ and affine surjection $\varphi : \mathbb{F}_2^n \to \mathbb{F}_2^m$ such that*

$$G = \varphi \circ F \circ \phi.$$

In this chapter we will assume that $m = n-1$. So our mappings from the previous definition are $\varphi : \mathbb{F}_2^n \to \mathbb{F}_2^{n-1}$ and $\phi : \mathbb{F}_2^{n-1} \to \mathbb{F}_2^n$.

Suppose we have a given vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. As n increases, finding all possible restrictions $G \prec F$, $G : \mathbb{F}_2^{n-1} \to \mathbb{F}_2^{n-1}$, becomes infeasible already for very small values of n. So we try to find some constraints for a more efficient approach.

Definition 36. *For an element $\alpha \in \mathbb{F}_2^n \setminus \{0\}$, we define as $\alpha^\perp$ the following set*

$$\alpha^\perp = \{x \in \mathbb{F}_2^n \mid \langle \alpha, x \rangle = 0\}.$$

We say that $\alpha^\perp$ is orthogonal of α. We denote complement of $\alpha^\perp$ as $\overline{\alpha^\perp}$ (the complement is the set $\mathbb{F}_2^n \setminus \alpha^\perp$).

The set $\alpha^\perp$ is a hyperplane, which means that $\alpha^\perp$ is of dimension $n-1$. The important fact is that for every hyperplane there is an element $\alpha \neq 0$ such that the hyperplane is of the form $\alpha^\perp$. This follows from the fact that we can find an orthogonal basis $\mathcal{B}$ of the hyperplane and then find α such that $\mathcal{B} \cup \{\alpha\}$ is an orthogonal basis of the whole vector space. The following definitions are given in [5, Section 3].

Definition 37. *Let $\beta \in \mathbb{F}_2^n$ and $\gamma \in \mathbb{F}_2^n$ such that $\langle \beta, \gamma \rangle = 1$. We define projection $\rho_\beta^{(\gamma)}$ of $\mathbb{F}_2^n$ to $\gamma^\perp$ as*

$$\begin{aligned} \rho_\beta^{(\gamma)} : \mathbb{F}_2^n &\to \gamma^\perp \\ x &\mapsto x + \beta \cdot \langle \gamma, x \rangle. \end{aligned}$$

The assumption $\langle \beta, \gamma \rangle = 1$ implies that $\beta \notin \gamma^\perp$. Since we mentioned that the orthogonal of an element is a $n-1$-dimensional linear hyperplane, we can represent $\mathbb{F}_2^n$ as the direct sum $\mathbb{F}_2^n = \gamma^\perp \oplus \{0, \beta\}$. Therefore any vector $x \in \mathbb{F}_2^n$ can be written as $x = x_\beta \oplus x_\gamma$, where $x_\beta \in \{0, \beta\}$ and $x_\gamma \in \gamma^\perp$. Using the projection to $\gamma^\perp$ we get $\rho_\beta^{(\gamma)}(x) = x_\gamma$. Thus with this projection we can define trim of a vectorial Boolean function.

Definition 38. *Let $n \in \mathbb{N}$, $n \geq 2$, $\beta \in \mathbb{F}_2^n$ be a non-zero element and let $H \subseteq \mathbb{F}_2^n$ be a hyperplane of dimension $n-1$, so that $H = \alpha^\perp$ or $H = \overline{\alpha^\perp}$ for some non-zero $\alpha \in \mathbb{F}_2^n$. Let $\epsilon \in \mathbb{F}_2^n$ be zero if $H = \alpha^\perp$ and $\epsilon \notin \alpha^\perp$ otherwise, and let $\gamma \in \mathbb{F}_2^n \backslash \beta^\perp$. The* trim of a function F along (H, β) with respect to ϵ, γ *is then defined as*

$$\begin{aligned} \mathcal{T}_{H \rightsquigarrow \beta}^{\epsilon,\gamma} F : \alpha^\perp &\to \gamma^\perp \\ x &\mapsto \rho_\beta^{(\gamma)} \circ F(x+\epsilon) = \begin{cases} F(x+\epsilon) & \text{if } F(x+\epsilon) \in \gamma^\perp \\ F(x+\epsilon) + \beta & \text{otherwise} \end{cases}. \end{aligned}$$

We can easily see, that the trim of a function is well defined, since

$$\mathcal{T}_{H \rightsquigarrow \beta}^{\epsilon,\gamma} F(x) = \rho_\beta^{(\gamma)} \circ F(x+\epsilon) = F(x+\epsilon) + \beta \cdot \langle \gamma, F(x+\epsilon) \rangle$$

and $\langle \gamma, F(x+\epsilon) \rangle = 0$ if and only if $F(x+\epsilon) \in \gamma^\perp$ and $\langle \gamma, F(x+\epsilon) \rangle = 1$ otherwise.

4.2 EA-equivalence of Trim Spectrum

In this section we start with three important statements. The first is that all trims of a function are affine-equivalent. In the second proposition we will prove that every restriction of the vectorial Boolean function F is EA-equivalent to a trim of F, and in the third we will show, that if two vectorial Boolean functions are EA-equivalent, then their trims are also EA-equivalent.

The following proposition and proof are based on Proposition 1 from [5, Section 3]. The proof is described in more detail in this thesis.

Proposition 25. *Let $n \in \mathbb{N}$, $n \geq 2$. For a fixed choice of (H, β) with $\beta \in \mathbb{F}_2^n \backslash \{0\}$ and $H \subseteq \mathbb{F}_2^n$ being a hyperplane of dimension $n-1$, all trims of F along (H, β) with respect to some ϵ, γ are affine-equivalent.*

Proof. We want to prove the statement for all possible combinations of ϵ and γ. Therefore we start by fixing γ. Since γ must satisfy the definition of trim of a function, we know that $\gamma \in \mathbb{F}_2^n \backslash \beta^\perp$. Suppose that $H = \alpha^\perp$ for some $\alpha \in \mathbb{F}_2^n$.

This implies from the definition of trim that we have only one possible choice for ϵ and that is $\epsilon = 0$. Thus for the fixed choice of β, $H = \alpha^\perp$ and γ we have only one possible trim.

Now let us assume that $H = \overline{\alpha^\perp}$. Thus we know that $\epsilon \notin \alpha^\perp$, which implies that $\epsilon \in \mathbb{F}_2^n \setminus \alpha^\perp$. So let us assume that we have $\epsilon, \epsilon' \in \mathbb{F}_2^n \setminus \alpha^\perp$. Notice that since $\mathbb{F}_2^n = \left(\mathbb{F}_2^n \setminus \alpha^\perp\right) \oplus \alpha^\perp$, then any element $x \in \mathbb{F}_2^n$ can be written as $x = x_1 \oplus x_2$, where $x_1 \in \mathbb{F}_2^n \setminus \alpha^\perp$ and $x_2 \in \alpha^\perp$. Therefore there exists element $\epsilon_\alpha \in \alpha^\perp$ such that $\epsilon' = \epsilon + \epsilon_\alpha$. Thus

$$\begin{aligned}
\mathcal{T}^{\epsilon',\gamma}_{H\rightsquigarrow\beta}F(x) &= \rho^{(\gamma)}_\beta \circ F(x+\epsilon') \\
&= F(x+\epsilon') + \beta\left\langle \gamma, F(x+\epsilon')\right\rangle \\
&= F(x+\epsilon_\alpha+\epsilon) + \beta\left\langle \gamma, F(x+\epsilon_\alpha+\epsilon)\right\rangle \\
&= F((x+\epsilon_\alpha)+\epsilon) + \beta\left\langle \gamma, F((x+\epsilon_\alpha)+\epsilon)\right\rangle \\
&= \rho^{(\gamma)}_\beta \circ F((x+\epsilon_\alpha)+\epsilon) \\
&= \mathcal{T}^{\epsilon,\gamma}_{H\rightsquigarrow\beta}F(x+\epsilon_\alpha).
\end{aligned}$$

This implies that $\mathcal{T}^{\epsilon',\gamma}_{H\rightsquigarrow\beta}F(x) = I_n \circ \mathcal{T}^{\epsilon',\gamma}_{H\rightsquigarrow\beta}F \circ A_1(x)$, where A_1 is an affine permutation such that $A_1(x) = I_n(x) + \epsilon_\alpha$, therefore $\mathcal{T}^{\epsilon',\gamma}_{H\rightsquigarrow\beta}F(x)$ is affine-equivalent to $\mathcal{T}^{\epsilon,\gamma}_{H\rightsquigarrow\beta}F(x+\epsilon_\alpha)$.

Now suppose we have fixed $\epsilon \in \mathbb{F}_2^n$. Suppose we have $\gamma, \gamma' \in \mathbb{F}_2^n \setminus \beta^\perp$. Let us define the following mapping Q as

$$\begin{aligned}
Q : \gamma^\perp &\to \gamma'^\perp \\
x &\mapsto x + \beta\left\langle \gamma', x\right\rangle
\end{aligned}$$

We want to prove that such a mapping is bijective and linear. For the bijection we start with the kernel of Q. Let us choose any $x \in \gamma^\perp$, thus $x + \beta\left\langle \gamma', x\right\rangle = 0 \iff x = \beta\left\langle \gamma', x\right\rangle$. Now we have two possible cases. First, if $\left\langle \gamma', x\right\rangle = 0$, then $x = 0$. Second, if $\left\langle \gamma', x\right\rangle = 1$, then $x = \beta$. But we know that $x \in \gamma^\perp \iff \left\langle x, \gamma\right\rangle = 0$, and since $x = \beta$ it must also hold that $\left\langle \beta, \gamma\right\rangle = 0$. But since $\gamma \in \mathbb{F}_2^n \setminus \beta^\perp$, it implies that $\left\langle \gamma, \beta\right\rangle = 1$, which is a contradiction. Thus $\operatorname{Ker}(Q) = \{0\}$, which implies that Q is injective. For the surjection let us assume that we have $y_1, y_2 \in \gamma^\perp$ such that $y_1 \neq y_2$ and $Q(y_1) = Q(y_2)$.

$$\begin{aligned}
Q(y_1) &= Q(y_2) \\
y_1 + \beta\left\langle \gamma', y_1\right\rangle &= y_2 + \beta\left\langle \gamma', y_2\right\rangle \\
(y_1+y_2) + \beta\left\langle \gamma', y_1+y_2\right\rangle &= 0 \\
Q(y_1+y_2) &= 0
\end{aligned}$$

We know from the $\operatorname{Ker}(Q) = \{0\}$ that $y_1 + y_2 = 0$, which implies that $y_1 = y_2$. This is a contradiction. Therefore Q is bijective.

For linearity we define a matrix Γ which has n rows and each row is equal to γ'. Therefore $\Gamma(x)$ has a value of either 0 or 1 in each row. This value is exactly the value of $\left\langle \gamma', x\right\rangle$. Let us also define the matrix $B = I_n(\beta)$. Therefore $B\Gamma(x) = \beta\left\langle \gamma', x\right\rangle$. Now let us look at $I_n(x) + B\Gamma(x)$. This is equal to $Q(x)$. Thus

$Q(x) = I_n(x) + B\Gamma(x) = (I_n + B\Gamma)(x)$, which is a linear mapping. Hence

$$\begin{aligned}Q \circ \mathcal{T}^{\epsilon,\gamma}_{H\rightsquigarrow\beta}F(x) \circ I_n(x) &= Q \circ \rho^{(\gamma)}_{\beta} \circ F(x+\epsilon)\\ &= Q \circ (F(x+\epsilon) + \beta \langle \gamma, F(x+\epsilon)\rangle)\\ &= F(x+\epsilon) + \beta\langle\gamma, F(x+\epsilon)\rangle + \beta\langle\gamma', F(x+\epsilon) + \beta\langle\gamma, F(x+\epsilon)\rangle\rangle\\ &= \begin{cases} F(x+\epsilon) + \beta\langle\gamma', F(x+\epsilon)\rangle\\ F(x+\epsilon) + \beta\langle\gamma', F(x+\epsilon)\rangle + \beta + \beta\langle\gamma',\beta\rangle\end{cases}\end{aligned}$$

where the first case occurs when $\langle\gamma, F(x+\epsilon)\rangle = 0$ and the second case occurs when $\langle\gamma, F(x+\epsilon)\rangle = 1$. Since $\gamma' \in \mathbb{F}_2^n \setminus \beta^{\perp}$ we know that $\langle\gamma', \beta\rangle = 1$, thus both cases are equal. Therefore

$$Q \circ \mathcal{T}^{\epsilon,\gamma}_{H\rightsquigarrow\beta}F(x) \circ I_n(x) = F(x+\epsilon) + \beta\langle\gamma', F(x+\epsilon)\rangle = \mathcal{T}^{\epsilon,\gamma'}_{H\rightsquigarrow\beta}F(x)$$

which implies that $\mathcal{T}^{\epsilon,\gamma}_{H\rightsquigarrow\beta}F(x)$ and $\mathcal{T}^{\epsilon,\gamma'}_{H\rightsquigarrow\beta}F(x)$ are linear-equivalent.

□

From the proposition, we know that the choice of ϵ and γ is arbitrary up to affine-equivalence (which is a special case of EA-equivalence due to Proposition 7). Therefore we can say that the EA-equivalence class of $\mathcal{T}^{\epsilon,\gamma}_{H\rightsquigarrow\beta}F$ is the trim of F along (H,β), denoted by $\mathcal{T}_{H\rightsquigarrow\beta}F$.

The following proposition and proof are based on Proposition 2 from [5, Section 3]. The proof is described in more detail in this thesis.

Proposition 26. *Let $n \in \mathbb{N}$, $n \geq 2$. Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $G : \mathbb{F}_2^{n-1} \to \mathbb{F}_2^{n-1}$ be two vectorial Boolean functions such that $G \prec F$. Then, there exists a $\beta \in \mathbb{F}_2^n \setminus \{0\}$ and an affine hyperplane $H \subseteq \mathbb{F}_2^n$ of dimension $n-1$ such that $G \approx_{\mathrm{EA}} \mathcal{T}_{H\rightsquigarrow\beta}F$.*

Proof. We have $G \prec F$, which means from the definition, that there exists an affine injective mapping $\phi : \mathbb{F}_2^{n-1} \to \mathbb{F}_2^n$ and an affine surjection $\varphi : \mathbb{F}_2^n \to \mathbb{F}_2^{n-1}$ such that

$$G = \varphi \circ F \circ \phi.$$

Since ϕ is an affine mapping, we can rewrite it as $\phi = \widetilde{\phi} + \epsilon$, where $\widetilde{\phi}$ is the linear part and $\epsilon \in \mathbb{F}_2^n$ is a constant.

Let us assume that $\widetilde{\phi} : \mathbb{F}_2^{n-1} \to \mathrm{Im}(\widetilde{\phi})$. We know that $\phi = \widetilde{\phi} + \epsilon$ is an injective mapping. Therefore $\widetilde{\phi}$ is also an injective mapping. It is also straightforward to see that $\forall k \in \mathrm{Im}(\widetilde{\phi})\ \exists l \in \mathbb{F}_2^{n-1}$ such that $l \mapsto k$. So all together we know that $\widetilde{\phi}$ is a linear bijection, thus an isomoprhism. Therefore $\mathrm{Im}(\widetilde{\phi})$ is a hyperplane, which implies that $\mathrm{Im}(\widetilde{\phi}) = \alpha^{\perp}$ for some non-zero $\alpha \in \mathbb{F}_2^n$. We can also rewrite the mapping φ. Suppose $\varphi = \widetilde{\varphi} + b$, where $\widetilde{\varphi}$ is the linear part of the mapping φ and $b \in \mathbb{F}_2^{n-1}$. The mapping $\widetilde{\varphi}$ is a surjection, since φ is surjective. Since $\widetilde{\varphi}(x)$ is a linear mapping, it can be represented as a $(n-1) \times n$ matrix over $\mathbb{F}_2$. Let the matrix be M, then $\forall x \in \mathbb{F}_2^n$: $\widetilde{\varphi}(x) = Mx$. The mapping $\widetilde{\varphi}$ is surjective, so $\mathrm{rank}(M) = n-1$. This means that the matrix M consists of a linearly independent sequence of column vectors with cardinality $n-1$. Suppose that P^{-1} is an $n \times n$ permutation matrix, permuting linearly independent $n-1$ columns of M to the left. Then we have

$$MP^{-1} = \begin{pmatrix} A^{-1} & \sigma \end{pmatrix},$$

where A is an invertible $(n-1) \times (n-1)$ matrix and σ is a column vector from $\mathbb{F}_2^{n-1}$, which is the remaining column from M. Now look at the mapping $G \circ \widetilde{\phi}^{-1} : \mathrm{Im}(\widetilde{\phi}) \to \mathbb{F}_2^{n-1}$. We need to rewrite this for future use.

$$\begin{aligned}
G \circ \widetilde{\phi}^{-1}(x) &= (\varphi \circ F \circ \phi) \circ \widetilde{\phi}^{-1}(x) \\
&= \varphi \circ F \circ \left(\widetilde{\phi} + \epsilon\right) \circ \widetilde{\phi}^{-1}(x) \\
&= \varphi \circ F \circ \left(\widetilde{\phi}\left(\widetilde{\phi}^{-1}(x)\right) + \epsilon\right) \\
&= \varphi \circ F(x + \epsilon) \\
&= (\widetilde{\varphi} + b) \circ F(x + \epsilon) \\
&= \widetilde{\varphi} \circ F(x + \epsilon) + b
\end{aligned}$$

Now we want to get the equation for the trim $\mathcal{T}_{H \rightsquigarrow \beta} F$ for some affine hyperplane $H \subseteq \mathbb{F}_2^n$ and $\beta \in \mathbb{F}_2^n$. We will use the following equality for a vector in $\mathbb{F}_2^n$. The further notation is not strict because it combines matrices, mappings and elements from $\mathbb{F}_2$ which are included in the matrix notation, but introducing some new notation including the dimensions of the elements would only be more confusing.

$$\begin{aligned}
\begin{pmatrix} A\left(G \circ \widetilde{\phi}^{-1}(x)\right) \\ 0 \end{pmatrix} &= \begin{pmatrix} A\left(\widetilde{\varphi} \circ F(x+\epsilon) + b\right) \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} A\left(M\left(F(x+\epsilon) + b\right)\right) \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} AM\left(F(x+\epsilon)\right) \\ 0 \end{pmatrix} + \begin{pmatrix} Ab \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} AMP^{-1}P\left(F(x+\epsilon)\right) \\ 0 \end{pmatrix} + \begin{pmatrix} Ab \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} AMP^{-1} \\ 0 \end{pmatrix} \begin{pmatrix} P\left(F(x+\epsilon)\right) \\ 0 \end{pmatrix} + \begin{pmatrix} Ab \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} A\left(A^{-1}\ \sigma\right) \\ 0 \end{pmatrix} \begin{pmatrix} P\left(F(x+\epsilon)\right) \\ 0 \end{pmatrix} + \begin{pmatrix} Ab \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} I_{n-1} & A\sigma \\ 0 & 0 \end{pmatrix} P\left(F(x+\epsilon)\right) + \begin{pmatrix} Ab \\ 0 \end{pmatrix}
\end{aligned}$$

The last equation can be rewritten using the vector e_n, which is a zero vector with 1 at the n-th position.

$$\begin{aligned}
\begin{pmatrix} A\left(G \circ \widetilde{\phi}^{-1}(x)\right) \\ 0 \end{pmatrix} &= P\left(F(x+\epsilon)\right) + \begin{pmatrix} A\sigma \\ 1 \end{pmatrix} \langle e_n, P\left(F(x+\epsilon)\right) \rangle + \begin{pmatrix} Ab \\ 0 \end{pmatrix} \\
&= P\left(F(x+\epsilon)\right) + \begin{pmatrix} A\sigma \\ 1 \end{pmatrix} \langle P^T e_n, F(x+\epsilon) \rangle + \begin{pmatrix} Ab \\ 0 \end{pmatrix} \\
&= P\left(F(x+\epsilon) + P^{-1}\begin{pmatrix} A\sigma \\ 1 \end{pmatrix} \langle P^T e_n, F(x+\epsilon) \rangle + P^{-1}\begin{pmatrix} Ab \\ 0 \end{pmatrix}\right)
\end{aligned}$$

Now we rearrange the last equation and multiply it by the P^{-1} matrix from the left. We then get

$$\begin{aligned}
P^{-1}\begin{pmatrix} A\left(G\circ\widetilde{\phi}^{-1}(x)\right) \\ 0 \end{pmatrix} + P^{-1}\begin{pmatrix} A\sigma \\ 0 \end{pmatrix} &= F(x+\epsilon) + P^{-1}\begin{pmatrix} A\sigma \\ 1 \end{pmatrix}\langle P^T e_n, F(x+\epsilon)\rangle \\
&= \mathcal{T}_{H\rightsquigarrow\beta}\, F(x)
\end{aligned}$$

for $\beta = P^{-1}\begin{pmatrix} A\sigma \\ 1 \end{pmatrix}$, $\gamma = P^T e_n$ and $H = \operatorname{Im}(\widetilde{\phi})$. Finally, we want to show, that G is EA-equivalent to the $\mathcal{T}_{H\rightsquigarrow\beta}\, F$. To do this, we use the last equation

$$\begin{aligned}
P^{-1}\begin{pmatrix} A\left(G\circ\widetilde{\phi}^{-1}(x)\right) \\ 0 \end{pmatrix} + P^{-1}\begin{pmatrix} A\sigma \\ 0 \end{pmatrix} &= \mathcal{T}_{H\rightsquigarrow\beta}\, F(x) \\
\begin{pmatrix} A\left(G\circ\widetilde{\phi}^{-1}(x)\right) \\ 0 \end{pmatrix} + \begin{pmatrix} A\sigma \\ 0 \end{pmatrix} &= P\,\mathcal{T}_{H\rightsquigarrow\beta}\, F(x) \\
\begin{pmatrix} A\left(G\circ\widetilde{\phi}^{-1}(x)\right) \\ 0 \end{pmatrix} &= P\,\mathcal{T}_{H\rightsquigarrow\beta}\, F(x) + \begin{pmatrix} A\sigma \\ 0 \end{pmatrix} \\
\begin{pmatrix} A & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} G\circ\widetilde{\phi}^{-1}(x) \\ 0 \end{pmatrix} &= P\,\mathcal{T}_{H\rightsquigarrow\beta}\, F(x) + \begin{pmatrix} A\sigma \\ 0 \end{pmatrix} \\
\begin{pmatrix} A^{-1} & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} A & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} G\circ\widetilde{\phi}^{-1}(x) \\ 0 \end{pmatrix} &= \begin{pmatrix} A^{-1} & 0 \\ 0 & 0 \end{pmatrix}\left(P\,\mathcal{T}_{H\rightsquigarrow\beta}\, F(x) + \begin{pmatrix} A\sigma \\ 0 \end{pmatrix}\right)
\end{aligned}$$

$$\begin{aligned}
\begin{pmatrix} I_{n-1} & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} G\circ\widetilde{\phi}^{-1}(x) \\ 0 \end{pmatrix} &= \begin{pmatrix} A^{-1} & 0 \\ 0 & 0 \end{pmatrix} P\,\mathcal{T}_{H\rightsquigarrow\beta}\, F(x) + \begin{pmatrix} A^{-1} & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} A & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} \sigma \\ 0 \end{pmatrix} \\
\begin{pmatrix} G\circ\widetilde{\phi}^{-1}(x) \\ 0 \end{pmatrix} &= \begin{pmatrix} A^{-1} & 0 \\ 0 & 0 \end{pmatrix} P\,\mathcal{T}_{H\rightsquigarrow\beta}\, F(x) + \begin{pmatrix} \sigma \\ 0 \end{pmatrix} \\
\begin{pmatrix} G(x) \\ 0 \end{pmatrix} &= \begin{pmatrix} A^{-1} & 0 \\ 0 & 0 \end{pmatrix} P\,\mathcal{T}_{H\rightsquigarrow\beta}\, F\circ\widetilde{\phi}(x) + \begin{pmatrix} \sigma \\ 0 \end{pmatrix}
\end{aligned}$$

The last step is to multiply both sides by a $(n-1)\times n$ matrix, which is composed of the matrix I_{n-1} and the last column is a zero vector. Therefore,

$$\begin{aligned}
G(x) &= \begin{pmatrix} I_{n-1} & 0 \end{pmatrix}\begin{pmatrix} A^{-1} & 0 \\ 0 & 0 \end{pmatrix} P\,\mathcal{T}_{H\rightsquigarrow\beta}\, F\circ\widetilde{\phi}(x) + \begin{pmatrix} I_{n-1} & 0 \end{pmatrix}\begin{pmatrix} \sigma \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} A^{-1} & 0 \end{pmatrix} P\,\mathcal{T}_{H\rightsquigarrow\beta}\, F\circ\widetilde{\phi}(x) + \sigma
\end{aligned}$$

From the last equation we can see that we have satisfied conditions from the definition of EA-equivalence, so we have proved that G is EA-equivalent to $\mathcal{T}_{H\rightsquigarrow\beta}$.

□

With this proposition, we can reduce the number of all restrictions for a given function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ since the trims depend only on H, β. For each non-zero element $\alpha \in \mathbb{F}_2^n$ we can define the hyperplane as either $\alpha^{\perp}$ or $\overline{\alpha^{\perp}}$. Thus we have $2(2^n-1)$ possible hyperplanes. The β is a non-zero element from $\mathbb{F}_2^n$. Therefore we have $2(2^n-1)^2$ possible choices.

Now we will establish another proposition, which stands for the fact that for two EA-equivalent functions, their trims are also EA-equivalent, i.e. for EA-equivalent functions $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$, the multiset of all possible trims of F is the same as the multiset of all possible trims of G.

The following proposition and proof are based on the proposition from [5, Section 3]. The proof is described in more detail in this thesis.

Proposition 27. *Let $n \in \mathbb{N}$, $n \geq 2$ and let $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be EA-equivalent via $G = A_2 \circ F \circ (A_1 + a_1) + a_2 + B$ with $A_1, A_2 \in \mathrm{GL}(n, \mathbb{F}_2)$, $a_1, a_2 \in \mathbb{F}_2^n$ and B being an affine function in $\mathbb{F}_2^n$. Then, for each hyperplane $H \subseteq \mathbb{F}_2^n$ and each $\beta \in \mathbb{F}_2^n \setminus \{0\}$, we have that $\mathcal{T}_{H \rightsquigarrow \beta}\, G$ is EA-equivalent to $\mathcal{T}_{H' \rightsquigarrow \beta'}\, F$, where $H' = A_1(H) + a_1$ and $\beta' = A_2^{-1}(\beta)$.*

Proof. Let us assume that $H = \alpha^\perp$ or $H = \overline{\alpha^\perp}$ for some non-zero $\alpha \in \mathbb{F}_2^n$. Let also $\gamma \in \mathbb{F}_2^n \setminus \beta^\perp$ and $\epsilon \in \mathbb{F}_2^n$ such that $\epsilon = 0$ if $H = \alpha^\perp$ and $\epsilon \notin \alpha^\perp$ if $H = \overline{\alpha^\perp}$. Let us also define the following notation for a $n \times n$ matrix A as follows

$$A(H) = \{A(x) \mid x \in H\}.$$

Now since $A_1 \in \mathrm{GL}(n, \mathbb{F}_2)$ we know that $A_1(H) = \alpha'^\perp$ for some $\alpha' \in \mathbb{F}_2^n$.

We will divide the proof into three cases. We start with the case that $G = F \circ (A_1 + a_1)$. Let us consider that $H = \alpha^\perp$. This implies that $\epsilon = 0$. Therefore

$$\begin{aligned}
\mathcal{T}^{0,\gamma}_{H \rightsquigarrow \beta}\, G(x) &= G(x) + \beta \langle \gamma, G(x) \rangle \\
&= F(A_1(x) + a_1) + \beta \langle \gamma, F(A_1(x) + a_1) \rangle \\
&= \mathcal{T}^{\epsilon',\gamma}_{H' \rightsquigarrow \beta}\, F(x')
\end{aligned}$$

Now we specify the new variables. For H' we know that $A_1(x) + a_1 \in A_1(H) + a_1$, therefore $H' := A_1(H) + a_1$. Also, if $a_1 \in A_1(H)$, then $H' = A_1(H) = \alpha'^\perp$ and if $a_1 \notin A_1(H)$, then $H' = A_1(H) + a_1 \neq \alpha'^\perp$. For the x' we know that $x' \in A_1(H) + a_1$, therefore we can define $x' := A_1(x) + a_1$. For the ϵ' we can use the fact that if $a_1 \in A_1(H)$, then $H' = \alpha'^\perp$, which implies that $\epsilon' = 0$. If $a_1 \notin A_1(H) = \alpha'^\perp$, then we can define $\epsilon' := a_1$. Therefore

$$\mathcal{T}^{\epsilon',\gamma}_{H' \rightsquigarrow \beta}\, F(x') = \begin{cases} \mathcal{T}^{0,\gamma}_{A_1(H)+a_1 \rightsquigarrow \beta}\, F(A_1(x) + a_1) & \text{if } a_1 \in A_1(H) \\ \mathcal{T}^{a_1,\gamma}_{A_1(H)+a_1 \rightsquigarrow \beta}\, F(A_1(x)) & \text{if } a_1 \notin A_1(H) \end{cases}$$

which implies that the trims are affine-equivalent.

Now let us consider $H = \overline{\alpha^\perp} = \mathbb{F}_2^n \setminus \alpha^\perp$. Therefore there exists $\epsilon \notin \alpha^\perp$ such that $H = \alpha^\perp + \epsilon$. Thus

$$\begin{aligned}
\mathcal{T}^{0,\gamma}_{H \rightsquigarrow \beta}\, G(x) &= G(x) + \beta \langle \gamma, G(x) \rangle \\
&= F(A_1(x + \epsilon) + a_1) + \beta \langle \gamma, F(A_1(x + \epsilon) + a_1) \rangle \\
&= F(A_1(x) + a_1 + A_1(\epsilon)) + \beta \langle \gamma, F(A_1(x) + a_1 + A_1(\epsilon)) \rangle \\
&= F(A_1(x) + (a_1 + A_1(\epsilon))) + \beta \langle \gamma, F(A_1(x) + (a_1 + A_1(\epsilon))) \rangle \\
&= \mathcal{T}^{\epsilon',\gamma}_{H' \rightsquigarrow \beta}\, F(x')
\end{aligned}$$

Now we set the new variables. For H' we know that since $A_1(x) + a_1 + A_1(\epsilon) \in A_1(H) + a_1 + A_1(\epsilon)$, therefore we can define $H' := A_1(H) + a_1 + A_1(\epsilon)$. If $a_1 + A_1(\epsilon) \in A_1(H)$, then $H' = A_1(H) = \alpha'^{\perp}$. This implies that $\epsilon' = 0$. If $a_1 + A_1(\epsilon) \notin A_1(H)$, then $H' = A_1(H) + a_1 + A_1(\epsilon)$. This implies that $\epsilon' := a_1 + A_1(\epsilon)$.

In both cases we can write H' as $H' = A_1(H) + a_1 + A_1(\epsilon)$, but since $A_1(\epsilon) \in A_1(H)$, we can write $H' = A_1(H) + a_1$.

If $\epsilon' = 0$, then $x' \in A_1(H) + a_1 + A_1(\epsilon)$, which implies that $x' := A_1(x) + a_1 + A_1(\epsilon)$. If $\epsilon' = a_1 + A_1(\epsilon)$, then $x' \in A_1(H)$, which implies that $x' := A_1(x)$.

$$\mathcal{T}^{\epsilon',\gamma}_{H'\rightsquigarrow\beta} F(x') = \begin{cases} \mathcal{T}^{0,\gamma}_{A_1(H)+a_1\rightsquigarrow\beta} F\left(A_1(x) + a_1 + A_1(\epsilon)\right) & \text{if } a_1 + A_1(\epsilon) \in A_1(H) \\ \mathcal{T}^{a_1+A_1(\epsilon),\gamma}_{A_1(H)+a_1\rightsquigarrow\beta} F\left(A_1(x)\right) & \text{if } a_1 + A_1(\epsilon) \notin A_1(H) \end{cases}$$

which implies that the trims are affine-equivalent.

For the second case let us assume that $G = A_2 \circ F + a_2$. Let us define $\gamma' := A_2^T(\gamma)$. Then for any $x \in \alpha^{\perp}$ we have

$$\begin{aligned} \mathcal{T}^{\epsilon,\gamma}_{H\rightsquigarrow\beta} G(x) &= G(x) + \beta \langle \gamma, G(x) \rangle \\ &= A_2\left(F(x+\epsilon)\right) + \beta \langle \gamma, A_2\left(F(x+\epsilon)\right) \rangle + a_2 + \beta \langle \gamma, a_2 \rangle \\ &= A_2\left(F(x+\epsilon)\right) + \beta \left\langle A_2^T(\gamma), F(x+\epsilon) \right\rangle + a_2 + \beta \langle \gamma, a_2 \rangle \\ &= A_2\left(F(x+\epsilon)\right) + \beta \langle \gamma', F(x+\epsilon) \rangle + a_2 + \beta \langle \gamma, a_2 \rangle \\ &= A_2\left(F(x+\epsilon) + A_2^{-1}(\beta) \langle \gamma', F(x+\epsilon) \rangle\right) + a_2 + \beta \langle \gamma, a_2 \rangle \\ &= A_2 \circ \mathcal{T}^{\epsilon,\gamma'}_{H\rightsquigarrow\beta'} F(x) + a_2 + \beta \langle \gamma, a_2 \rangle . \end{aligned}$$

which means that the trims are EA-equivalent. For the third case let us assume that $G = F + B$ and let $x \in \alpha^{\perp}$. Then

$$\begin{aligned} \mathcal{T}^{\epsilon,\gamma}_{H\rightsquigarrow\beta} G(x) &= G(x) + \beta \langle \gamma, G(x) \rangle \\ &= F(x+\epsilon) + B(x+\epsilon) + \beta \langle \gamma, F(x+\epsilon) + B(x+\epsilon) \rangle \\ &= F(x+\epsilon) + \beta \langle \gamma, F(x+\epsilon) \rangle + B(x+\epsilon) + \beta \langle \gamma, B(x+\epsilon) \rangle \\ &= \mathcal{T}^{\epsilon,\gamma}_{H\rightsquigarrow\beta} F(x) + \left(B(x+\epsilon) + \beta \langle \gamma, B(x+\epsilon) \rangle\right) . \end{aligned}$$

The element in the last brackets is an affine function. This implies that the trims are EA-equivalent.

All together we get in each of the three cases that the trims of a function are EA-equivalent. Since EA-equivalence is an equivalence relation, we can combine these cases and still get EA-equivalence of the trims.

□

Because of the Proposition 27 we can define the trim spectrum of a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ which is an EA-invariant, but first we introduce the following notation. The following definitions are from [5, Section 3].

Definition 39. *Let $\mathcal{H}_n$ be defined as the set of all $n-1$-dimensional hyperplanes of $\mathbb{F}_2^n$. That means*

$$\mathcal{H}_n = \{\alpha^{\perp} \mid \alpha \in \mathbb{F}_2^n \setminus \{0\}\} \cup \{\overline{\alpha^{\perp}} \mid \alpha \in \mathbb{F}_2^n \setminus \{0\}\}.$$

Definition 40. *Let $n \in \mathbb{N}$, $n \geq 2$. The* trim spectrum *of a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is the multiset of all trims of F along (H, β), where (H, β) takes all $2(2^n - 1)^2$ possibilities, i.e., the multiset $\{\mathcal{T}_{H \rightsquigarrow \beta} F \mid H \in \mathcal{H}_n, \beta \in \mathbb{F}_2^n \setminus \{0\}\}$.*

The following proposition and proof are based on the proposition from [5, Section 3]. The proof is described in more detail in this thesis.

Proposition 28. *The trim spectrum of a vectorial Boolean function is an EA-invariant. In other words, for $n \in \mathbb{N}$, $n \geq 2$, if $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ are EA-equivalents, the multisets $\{\mathcal{T}_{H \rightsquigarrow \beta} F \mid H \in \mathcal{H}_n, \beta \in \mathbb{F}_2^n \setminus \{0\}\}$ and $\{\mathcal{T}_{H \rightsquigarrow \beta} G \mid H \in \mathcal{H}_n, \beta \in \mathbb{F}_2^n \setminus \{0\}\}$ consist of the same EA-equivalence classes with the same multiplicities.*

Proof. We assume that the functions F and G are EA-equivalent. Therefore we know from the definition that there exist matrices $A_1, A_2 \in \mathrm{GL}(n, \mathbb{F}_2)$, elements $a_1, a_2 \in \mathbb{F}_2^n$ and affine function $B : \mathbb{F}_2^n \to \mathbb{F}_2^n$ such that

$$G = A_2 \circ F \circ (A_1 + a_1) + a_2 + B.$$

From the Proposition 27 we know that for any hyperplane $H \subseteq \mathbb{F}_2^n$ and any $\beta \in \mathbb{F}_2^n \setminus \{0\}$ the $\mathcal{T}_{H \rightsquigarrow \beta} G$ is EA-equivalent to $\mathcal{T}_{H' \rightsquigarrow \beta'} F$, where $H' = A_1(H) + a_1$ and $\beta' = A_2^{-1}(\beta)$. Now we define the following two mappings. The first one is

$$\begin{aligned} \psi : \mathcal{H}_n &\to \mathcal{H}_n \\ H &\mapsto H' = A_1(H) + a_1 \end{aligned}$$

and the second one is

$$\begin{aligned} \pi : \mathbb{F}_2^n \setminus \{0\} &\to \mathbb{F}_2^n \setminus \{0\} \\ \beta &\mapsto \beta' = A_2^{-1}(\beta) \end{aligned}$$

Since $A_1, A_2 \in \mathrm{GL}(n, \mathbb{F}_2)$ we know that the mappings ψ and π are bijections. Therefore

$$\{\mathcal{T}_{H' \rightsquigarrow \beta'} F \mid H' \in \mathcal{H}_n, \beta' \in \mathbb{F}_2^n \setminus \{0\}\} = \{\mathcal{T}_{H \rightsquigarrow \beta} F \mid H \in \mathcal{H}_n, \beta \in \mathbb{F}_2^n \setminus \{0\}\}.$$

The cardinality of the multisets from the statement of this proposition is the same because ψ, π are bijections, and the multisets consist of the same EA-equivalence classes because of the EA-equivalence of $\mathcal{T}_{H \rightsquigarrow \beta} G$ and $\mathcal{T}_{H' \rightsquigarrow \beta'} F$.

□

4.3 Algorithm

In this section we will present an algorithm to find the trim spectrum of a given function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and to find APN function among the functions in the trim spectrum. This algorithm will also convert the APN trim of a function (which is a function from $\alpha^\perp$ to $\gamma^\perp$) to a function $G : \mathbb{F}_2^{n-1} \to \mathbb{F}_2^{n-1}$. The implementation of the algorithm is attached in A.11.

4.3.1 Description

We start with the input to the algorithm. The input function can be any vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. From the theory in the previous sections we know that we can apply EA-equivalence. If two functions $F_1, F_2 : \mathbb{F}_2^n \to \mathbb{F}_2^n$ are EA-equivalent, then we know from Proposition 27 that their trim spectra consist of the same EA-equivalence classes. Thus, when considering functions up to EA-equivalence, it is sufficient to give only EA-inequivalent functions as input to the algorithm.

In our case, the input functions will always be APN, since these functions are in our main focus in this thesis.

The main function of the algorithm is `Finding_Trim_Spectrum`. The function follows the theory from the previous sections of this chapter. From Proposition 25 we know that for a fixed choice of (H, β) all trims of F along (H, β) are affine-equivalent with respect to some ϵ, γ. This implies that for fixed (H, β) it is sufficient to consider only one choice of ϵ and γ.

From the Proposition 26 we know that for every $G \prec F$ there exists (H, β) such that G is EA-equivalent to $\mathcal{T}_{H \rightsquigarrow \beta} F$. So if we consider all possible (H, β), then we have examined all such $G : \mathbb{F}_2^{n-1} \to \mathbb{F}_2^{n-1}$ up to EA-equivalence. We know that the cardinality of the trim spectrum is $2(2^n - 1)$, thus we have $2(2^n - 1)$ possible choices for (H, β).

The function is therefore divided into two cases based on the definition of the hyperplane H via the line 4. In the first iteration of the `For` loop we assume that $H = \alpha^\perp$, and in the second iteration of the `For` loop we assume that $H = \overline{\alpha^\perp}$.

If $H = \alpha^\perp$, then we know from the definition of a trim of a function that $\epsilon = 0$. If $H = \overline{\alpha^\perp}$, then $\epsilon \in \overline{\alpha^\perp}$, so we have to find such ϵ with the `while` loop on the line 8, which is only executed if $H = \overline{\alpha^\perp}$. Also from the definition we know that $\gamma \in \mathbb{F}_2^n \setminus \beta^\perp$, so we need to find such γ in a `while` loop on the line 14.

```
 1: function FINDING_TRIM_SPECTRUM(“F”,$n$):
 2:     $\epsilon \leftarrow 0$
 3:     “trim_spectrum” ← initialised as an empty array
 4:     for $H \in [0, 1]$ do
 5:         for $\alpha \in [1, \ldots, 2^n - 1]$ do
 6:             if $H = 1$ then
 7:                 $\epsilon \leftarrow 1$
 8:                 while $\langle \epsilon, \alpha \rangle = 0$ do
 9:                     $\epsilon \leftarrow \epsilon + 1$
10:                 end while
11:             end if
12:             for $\beta \in [1, \ldots, 2^n - 1]$ do
13:                 $\gamma \leftarrow 1$
14:                 while $\langle \beta, \gamma \rangle = 0$ do
15:                     $\gamma \leftarrow \gamma + 1$
16:                 end while
17:                 “trim” ← initialised as an empty array
18:                 for $x \in [0, \ldots, 2^n - 1]$ do
19:                     if $\langle \alpha, x \rangle = 0$ then
20:                         if $\langle \gamma, \mathrm{F}[x \oplus \epsilon] \rangle = 0$ then
```

```
                            trim[$x$]←F[$x \oplus \epsilon$]
                        **else**
                            trim[$x$]←F[$x \oplus \epsilon$]$\oplus\beta$
                        **end if**
                    **end if**
                    **if** $\langle \alpha, x\rangle = 1$ **then**
                        trim[$x$]← *init_value*
                    **end if**
                **end for**
                "trim_spectrum"←append "trim" into the list
                "parameters"←append [$\alpha, \beta, \gamma, \epsilon$] into the list
            **end for**
        **end for**
    **end for**
    **return**("trim_spectrum", "parameters")
**end function**
```

The output of the function is a set of look-up tables of all functions from the trim spectrum of F and a list of parameters $\alpha, \beta, \gamma, \epsilon$ used to generate a trim. Note that the look-up tables contain 2^n values, of which 2^{n-1} are *init_value*, since the trim of a function is only defined on 2^{n-1} elements. These look-up tables can still be used to determine whether the trim of a function is APN or not. For this we use the function `Is_Function_APN_For_Trim`, which calculates the DDT table, using only the values for which is the trim of a function is defined, and checks if all the values in the DDT are less or equal to 2.

Since we want to continue working with the APN trims we have obtained, we need to transform the trim (as a function from $\alpha^\perp$ to $\gamma^\perp$) into the function G from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2^{n-1}$. To do this, we need to find two mappings $\varphi : \mathbb{F}_2^{n-1} \to \alpha^\perp$ and $\pi : \gamma^\perp \to \mathbb{F}_2^{n-1}$, such that $G = \pi \circ \mathcal{T}_{H \rightsquigarrow \beta} F \circ \varphi$.

To do this, we implemented the function `Find_Function_G` which we will now describe.

The $\mathbb{F}_2^{n-1}, \alpha^\perp$ and $\gamma^\perp$ are vector spaces of dimension $n-1$, so we can find bases of each of them with cardinality $n-1$. For $\mathbb{F}_2^{n-1}$ we can take the basis $\{2^0, 2^1, 2^2, \ldots, 2^{n-2}\}$, where these integers represent elements of $\mathbb{F}_2^{n-1}$ via their binary notation as mentioned in subsection 1.2.3. For $\alpha^\perp$ and $\gamma^\perp$ we will use the following procedure. Without loss of generality we will only describe it for $\alpha^\perp$.

We start by finding all elements of the vector space $\alpha^\perp$, since $a_\alpha \in \alpha^\perp \iff \langle a_\alpha, \alpha\rangle = 0$ using the function `Find_All_Elements_of_Hyperplane`. Then we use the recursive function `Find_Basis_of_Hyperplane` we select from these elements a linear independent sequence of $n-1$ elements $v_{0,\alpha}, v_{1,\alpha}, \ldots, v_{n-2,\alpha}$ which is the basis of $\alpha^\perp$.

Now we can use the function `Find_Mapping`, which is based on the following idea. Since we want φ and π to be isomorphisms of vector spaces, we can define these mappings using the basis elements. Again, without loss of generality, we will only describe this for the mapping φ.

Let us define $\varphi(2^i) = v_{i,\alpha}$ for $i \in \{0, \ldots, n-2\}$. Then for any $a \in \mathbb{F}_2^{n-1}$ we can take the binary notation $[a_0, a_1, \ldots, a_{n-2}]$, thus $a = a_0 2^0 + a_1 2^1 + a_2 2^2 + \cdots +$

$a_{n-2}2^{n-2}$, therefore

$$\begin{aligned}\varphi(a) &= \varphi(a_0 2^0 + \cdots + a_{n-2}2^{n-2})\\ &= a_0\varphi(2^0) + \cdots + a_{n-2}\varphi(2^{n-2})\\ &= a_0 v_{0,\alpha} + \cdots + a_{n-2}v_{n-2,\alpha}\\ &= a_\alpha,\end{aligned}$$

where $a_\alpha \in \alpha^\perp$. so the output of the function `Find_Mapping` on the line 7 is a look-up table for φ, respectively for π on the line 8.

```
1: function FIND_FUNCTION_G($\alpha, \gamma$,"trim",$m$):
2:     $n \leftarrow m + 1$
3:     "elements_of_$\alpha^\perp$"$\leftarrow$Find_All_Elements_of_Hyperplane($\alpha, n$)
4:     "elements_of_$\gamma^\perp$"$\leftarrow$Find_All_Elements_of_Hyperplane($\gamma, n$)
5:     "basis_of_$\alpha^\perp$" $\leftarrow$ Find_Basis_of_Hyperplane("elements_of_$\alpha^\perp$",$m$)
6:     "basis_of_$\gamma^\perp$" $\leftarrow$ Find_Basis_of_Hyperplane("elements_of_$\gamma^\perp$",$m$)
7:     "mapping_for_$\alpha$" $\leftarrow$ Find_Mapping("basis_of_$\alpha^\perp$",$m$)
8:     "mapping_for_$\gamma$" $\leftarrow$ Find_Mapping("basis_of_$\gamma^\perp$",$m$)
9:     "lut" $\leftarrow$ initialised as an empty array
10:    for $\alpha \in [0, \dots, 2^m - 1]$ do
11:        $output\_of\_trim$ $\leftarrow$trim[mapping_for_$\alpha$[1][$i$]]
12:        $index \leftarrow 1$
13:        while mapping_for_$\gamma$[1][$index$] $\neq$ $output\_of\_trim$ & $index < 2^m - 1$ do
14:            $index \leftarrow index + 1$
15:        end while
16:        lut[$i$]$\leftarrow$ $index$
17:    end for
18:    return("lut")
19: end function
```

Finally we can formulate the Algorithm 4 which, for each input function, computes its trim spectrum and, if there is an APN function among the trims in the trim spectrum, computes the function $G : \mathbb{F}_2^{n-1} \to \mathbb{F}_2^{n-1}$.

4.3.2 Results

The following results corresponds to some results of [5, Section 3.1] and are attached in A.12.

From $n = 7$ to 6

First we try to replicate the results of [5, Section 3]. Let us denote $\mathcal{F} = \{F_1, \dots, F_{488}\}$ which is the set of all 7-bit quadratic APN functions founded in the [13]. All of these functions are pairwise EA-inequivalent as it is stated in [5]. The values of these functions can be found in [14] in a file `sevenBitAPN.py`. For these functions we run Algorithm 4.

The output of the algorithm is 438 APN functions $\mathcal{G} = \{G_1, \dots, G_{438}\}$. So for 50 functions of $\mathcal{F}$ there is no APN function in their trim spectrum. All of these functions are quadratic, which was tested using the result of Proposition 13.

Algorithm 4 Search for APN function in the trim spectrum

Input: a list “functions” of APN vectorial Boolean functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$
Output: a list of APN trims of an input functions and a list of the corresponding functions from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2^{n-1}$

```
1: for $i \in [0, \ldots,$ number of input functions] do
2:     $trim\_is\_APN \leftarrow 0$
3:     “trim_spectrum”, “parameters”$\leftarrow$Finding_Trim_Spectrum(functions[$i$],$n$)
4:     while ($k <$ number of trims in trim spectrum) & ($trim\_is\_APN = 0$) do
5:         $trim\_is\_APN$ $\leftarrow$Is_Function_APN_For_Trim(trim_spectrum[$k$],$n$)
6:         if $trim\_is\_APN$ =True then
7:             “list_of_APN_trims”$\leftarrow$append “trim_spectrum[$k$]” into the list
8:             “list_of_functions_G”$\leftarrow$append
9: Find_Function_G($\alpha, \gamma$, trim_spectrum[$k$],$n-1$) into the list
10:        end if
11:        $k \leftarrow k + 1$
12:    end while
13: end for
```

Using the Algorithm 2 (A.6) we classified functions from $\mathcal{G}$ into 11 EA-equivalence classes. This implies from the results of 2.3.3 that functions from two EA-equivalence classes are not in the trim spectra of some 7-bit quadratic APN function. Those two 2 EA-equivalence classes are classes 10 and 13 from the results A.8 for $n = 6$.

From $n = 6$ to 5

First we used as input representatives of classes 10 and 13 from the results A.8 for $n = 6$ (we choose first function from these classes in A.8). We get that the representative of the class 10 has a quadratic APN function in its trim spectrum and the representative of the class 13 does not have such function in its trim spectrum.

Next we have used 11 pairwise EA-inequivalent functions from $\mathcal{G}$ as an input. We get 9 APN functions $\mathcal{V} = \{V_1, \ldots, V_9\}$. All of these functions are quadratic, which was tested using the result of Proposition 13. Using the modified Algorithm 2 (A.7) we put the functions from $\mathcal{V}$ into 2 EA-equivalence classes. The trim of representative of the class 10 is EA-equivalent to one of them since we have only two EA-equivalence classes in dimension 5 (see Subsection 2.3.3).

From $n = 5$ to 4

We have used the functions V_1, V_2 as an input to the algorithm. We get 2 APN functions $\mathcal{W} = \{W_1, W_2\}$. All of these functions are quadratic, which was tested using the result of Proposition 13. Since in dimension 4 exists only one EA-equivalence class (see Subsection 2.3.3), the functions from $\mathcal{W}$ are EA-equivalent.

5. Finding Quadratic APN Functions with Maximum Linearity

This focuses on the last method for finding APN functions presented in this thesis. Based on the quadratic APN function $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$, we will construct a quadratic APN function $T : \mathbb{F}_2^{n+1} \to \mathbb{F}_2^{n+1}$ with maximum linearity. In the this chapter we will use the information provided by [5, Section 5].

Our contribution is to present the theory in a clearer way, to prove the statements in more detail, to use the provided pseudo-code of the implementation of Algorithm 1 from [5] to present our own Python implementation, to describe the algorithm in detail and in the context of the theory of this chapter and to replicate some of the results presented in [5, Section 5] and make them available in A.

5.1 Ortho-derivative of a Function

We start with the following definition which is used in Chapter 2 and in this chapter. The definition can be found in [5, Section 2].

Definition 41. *Let $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN function. The* ortho-derivative *of G is defined as the unique function $\pi_G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ with $\pi_G(0) = 0$ such that for all $\alpha \in \mathbb{F}_2^n \setminus \{0\}$, we have $\pi_G(\alpha) \neq 0$ and*

$$\forall x \in \mathbb{F}_2^n : \langle \pi_G(\alpha), B_\alpha(x) \rangle = 0,$$

where $B_\alpha : \mathbb{F}_2^n \to \mathbb{F}_2^n$, $x \mapsto G(x) + G(x+\alpha) + G(\alpha) + G(0)$.

The definition states that the ortho-derivative is well defined and unique. We will show, that this is true, but first we make an observation about B_α in the following lemma. This lemma is based on the statement in the proof of Theorem 1 in [5, Section 5], where it is stated without proof.

Lemma 29. *Let $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN vectorial Boolean function. Then the mapping $B_\alpha : \mathbb{F}_2^n \to \mathbb{F}_2^n$, $x \mapsto G(x) + G(x+\alpha) + G(\alpha) + G(0)$ is linear.*

Proof. Since G is a vectorial Boolean function, we can write it in the ANF as follows

$$G(x) = \sum_{u \in \mathbb{F}_2^n} a_u \prod_{i=1}^{n} x_i^{u_i},$$

where $a_u \in \mathbb{F}_2^n$, $x = (x_1, \ldots, x_n)$ and $u = (u_1, \ldots, u_n)$. We will divide the proof into three cases based on the degree of an element which is contained in the ANF. Let us start with an element of degree zero. Thus, u is a zero vector and the sum in the ANF expression of G contains $a_0 \in \mathbb{F}_2^n$. This means that B_α contains $a_0 + a_0 + a_0 + a_0$ which is a zero vector.

The next case is that the ANF contains an element of degree 1. Therefore, for some fixed u (u must be a vector composed of zeros and u_j such that $u_j = 1$, where $j \in \{1, \dots, n\}$), the sum in the ANF expression of G contains $a_u x_j^{u_j} = a_u x_j$. Thus, B_α contains $a_u x_j + a_u(x_j + \alpha_j) + a_j 0 + a_j \alpha_j = a_u x_j + a_u x_j + a_u \alpha_j + a_u \alpha_j = 0$. So the degree 1 element of G disappears in B_α.

The remaining case is that where the ANF contains an element of degree 2 (since G is quadratic, it cannot contain an element of higher degree). Therefore, for some fixed u (u must be a vector composed of zeros and u_j, u_k such that $u_j = u_k = 1$, where $j, k \in \{1, \dots, n\}$ and $j \neq k$), the sum in the ANF expression of G contains $a_u x_j^{u_j} x_k^{u_k} = a_u x_j x_k$. Thus B_α contains

$$\begin{aligned} &a_u x_j x_k + a_u(x_j + \alpha_j)(x_k + \alpha_k) + a_u 0 + a_u \alpha_j \alpha_k \\ &= a_u(x_j x_k + x_j x_k + x_j \alpha_x + x_k \alpha_j + \alpha_j \alpha_k + \alpha_j \alpha_k \\ &= a_u x_j \alpha_k + a_u x_k \alpha_j, \end{aligned}$$

which is an element of degree 1.

Therefore, we can see, that B_α does not contain an element of degree less than or greater than one, thus B_α is a linear mapping.

□

We can see from the proof, that also $G(x) + G(x + \alpha)$ is also a linear mapping. This implies that the image of mapping the $G(x) + G(x + \alpha)$ is a linear subspace of $\mathbb{F}_2^n$.

The function B_α is defined for given $\alpha \in \mathbb{F}_2^n \setminus \{0\}$. So let us fix such α. Now we are interested in the cardinality of the set

$$\{\beta \mid x \in \mathbb{F}_2^n, G(x) + G(x + \alpha) = \beta\}. \tag{5.1}$$

The function G is defined on the whole vector space $\mathbb{F}_2^n$, so every element of $\mathbb{F}_2^n$ is mapped somewhere. Also, since G is APN, we know that for every such β, if $G(x_1) + G(x_1 + \alpha) = \beta$, then for $x_2 := x_1 + \alpha$ we know that $G(x_2) + G(x_2 + \alpha) = \beta$. From the APN property we also know that there are no more such x that satisfy the equation for given β. Since the cardinality of $\mathbb{F}_2^n$ is 2^n, we know that if we divide the $\mathbb{F}_2^n$ into sets $\{x, x + \alpha\}$ such that these sets are disjoint, we have 2^{n-1} of such sets. Therefore the cardinality of the set in 5.1 is also 2^{n-1}. This implies that the cardinality of the images of B_α is also 2^{n-1}, since $G(0) + G(\alpha)$ is constant for given α.

All in all we have that the image of B_α is a linear subspace and the cardinality of this subspace is 2^{n-1}. This implies that we can find an orthogonal basis $v_1, \cdots, v_{n-1}$ of the subspace B_α. Such a basis can be transformed into an orthogonal basis of $\mathbb{F}_2^n$ by finding a unique vector $v_n \in \mathbb{F}_2^n$ such that $\langle v_n, v_i \rangle = 0$ for $i \in \{1, \dots, n - 1\}$.

We can now define $\pi_G(\alpha) = v_n$ and since such $v_n \in \mathbb{F}_2^n$ exists for all $\alpha \in \mathbb{F}_2^n \setminus \{0\}$ and is unique, we have shown that the ortho-derivative is also well-defined and unique.

The next important property of the ortho-derivatives is the following. The proof of the proposition can be found in [12].

Proposition 30. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be two EA-equivalent quadratic APN functions. Then their ortho-derivatives π_F and π_G are linear-equivalent.*

From the Proposition 7 we know that the Proposition 30 implies that the ortho-derivatives π_F and π_G are EA-equivalent.

5.2 EA-equivalence of Quadratic Boolean Functions

For the following section we need to prove that every quadratic Boolean function can be expressed in a certain form given by Theorem 32. We start with Theorem 31. Theorem 31 is based on part (2) of Dickson's theorem from [17, Chapter 15]. The proof of the theorem is technical and well presented in the cited source.

Let us start by noting that from the ANF of a Boolean function we can see that a quadratic Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ can be also expressed as

$$f(x) = x^T Q x + L(x) + \epsilon,$$

where Q is an $n \times n$ upper triangular matrix over $\mathbb{F}_2$ with zero diagonal and L is linear function from $\mathbb{F}_2^n \to \mathbb{F}_2$ and $\epsilon \in \mathbb{F}_2$.

Theorem 31. *A quadratic Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ such that*

$$f(x) = x^T Q x + L(x) + \epsilon,$$

where Q is an upper triangular matrix with zero diagonal, L is an arbitrary linear function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$ and $\epsilon \in \mathbb{F}_2$, is linear-equivalent to

$$h(y) = \sum_{i=1}^{r} y_{2i-1} y_{2i} + L_1(y) + \epsilon$$

where $2r = \operatorname{rank}(B)$ for $B = Q + Q^T$.

Using the form of a quadratic Boolean function given by Theorem 31, we can state the following theorem, which gives us the form of a quadratic Boolean function up to EA-equivalence.

Theorem 32. *Any quadratic Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$, such that $f(x) = x^T Q x + L(x) + \epsilon$, where Q is a upper triangular matrix with zero main diagonal, $L : \mathbb{F}_2^n \to \mathbb{F}_2$ is a linear function and $\epsilon \in \mathbb{F}_2$, is EA-equivalent to the function*

$$g(y) = \sum_{i=1}^{r} y_{2i-1} y_{2i},$$

where $2r = \operatorname{rank}(Q + Q^T)$.

Proof. From Theorem 31 we know that $f(x)$ is linear-equivalent to

$$h(y) = \sum_{i=1}^{r} y_{2i-1} y_{2i} + L_1(y) + \epsilon.$$

Thus for an affine function $B(y)$ defined as $B(y) := L_1(y) + \epsilon$ we have that

$$h(y) + B(y) = \sum_{i=1}^{r} y_{2i-1} y_{2i},$$

thus $f \approx_{\text{EA}} g$, where $g(y) := \sum_{i=1}^{r} y_{2i-1} y_{2i}$.

□

5.3 EA-equivalence of Quadratic Boolean Function with Maximum Linearity

Our goal in this section is to find for a quadratic vectorial Boolean function $F : \mathbb{F}_2^{n+1} \to \mathbb{F}_2^{n+1}$ EA-equivalent function T which has a certain form using linear functions $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$. Then we state conditions under which the function T (and hence the function F) is APN. Finally, we prove that a certain choice of a linear function L' holds EA-equivalent function T' to T. Note to the reader that we use the notation as in [5], where (x, y), for $x \in \mathbb{F}_2^n$ and $y \in \mathbb{F}_2$, usually denotes the element from $\mathbb{F}_2^{n+1}$.

We start by proving the Theorem 36, which is based on Proposition 4 from [5, Section 5]. For this we need to state and prove Lemmas 33 and 34 which are necessary for the proof of Proposition 35 which then leads to the proof of the theorem.

The statement of the following lemma is given without proof in the proof of Proposition 4 in [5, Section 4].

Lemma 33. *Let $T : \mathbb{F}_2^{n+1} \to \mathbb{F}_2^{n+1}$ be a vectorial Boolean function. Let t_i be the i-th coordinate function of T for $i \in \{1, \dots, n+1\}$. Then T is quadratic if and only if for any $i \in \{1, \dots, n+1\}$ function $x \mapsto t_i(x, 0) + t_i(x, 1)$ is of degree at most 1, where $x \in \mathbb{F}_2^n$.*

Proof. For the first implication, let us assume that T is quadratic. This implies that it consists of some quadratic coordinate functions. Let us take $i \in \{1, \dots, n+1\}$ such that t_i is a quadratic coordinate function. Such a function can be expressed by its ANF. Hence

$$t_i(z) = \sum_{\mathrm{wt_H}(u) \leq 2,\ u \in \mathbb{F}_2^{n+1}} b_{i,u} \prod_{i=1}^{n+1} z_i^{u_i},$$

where $b_{i,u} \in \mathbb{F}_2$ and $z = (z_1, \dots, z_{n+1})$. This can be rearranged so that

$$t_i(z) = \sum_{\mathrm{wt_H}(u) \leq 2,\ u_{n+1}=0} b_{i,u} \prod_{i=1}^{n+1} z_i^{u_i} + \sum_{\mathrm{wt_H}(u) \leq 2,\ u_{n+1}=1} b_{i,u} \prod_{i=1}^{n+1} z_i^{u_i}.$$

Since the inputs $(x, 0)$ and $(x, 1)$ differ only in the $n+1$-th coordinate, we can see that the first sum appears in both $t_i(x, 0)$ and $t_i(x, 1)$. Therefore, we are only interested in the elements of the second sum.

For $t_i(x, 0)$ we can see that if $u_{n+1} = 1$, then $z_{n+1}^{u_{n+1}} = 0^1 = 0$, which implies that there is no non-zero element in the second sum for $t_i(x, 0)$.

For $t_i(x, 1)$ we can divide the second sum based on the Hamming weight. If $\mathrm{wt_H}(u) = 2$, then we have some index $j \neq n+1$ such that $u_j = 1$, so we have the product $z_j^{u_j} z_{n+1}^{u_{n+1}}$ which for $(x, 1)$ is equal to $x_j^{u_j} 1 = x_j$, which is of degree one.

If $\mathrm{wt_H}(u) = 1$, then we have only one possible choice of u, and that is $u = (0, \dots, 0, 1)$, so $z_{n+1}^{u_{n+1}} = 1$, which is of degree 0. For $\mathrm{wt_H}(u) = 0$, there is no such element u such that $u_{n+1} = 1$. This means that we have proved that $t_i(x, 0) + t_i(x, 1)$ is of degree at most one.

Now for the second implication. Let us assume for the sake of contradiction that the function T is of degree higher than 2. Without loss of generality let us assume that it is of degree 3. Then there exists a coordinate function t_i which is of degree 3. Therefore there exists $u \in \mathbb{F}_2^{n+1}$ such that $u_j = u_k = u_l = 1$, where $j, k, l \in \{1, \dots, n+1\}$ and j, k, l are distinct. Since T is arbitrary, we can assume that $l = n+1$. So for such u we have that $t_i(z)$ contains $b_{i,u} z_j z_k z_{n+1}$. This implies that $t_i(x, 0)$ contains $b_{i,u} x_j x_k 0 = 0$ and $t_i(x, 1)$ contains $b_{i,u} x_j x_k$. Thus, $t_i(x, 0) + t_i(x, 1)$ contains $b_{i,u} x_j x_k$, which is an element of degree 2, contradicting the assumption that the function $x \mapsto t_i(x, 0) + t_i(x, 1)$ is of degree at most 1.

□

Lemma 34. *Let $x \in \mathbb{F}_2^n$ be such that $x = (x_1, \dots, x_n)$ and $l \in \{1, \dots, n\}$. Let $\ell(x_1, \dots, x_{l-1}, x_{l+1}, \dots, x_n)$ be a non-zero linear function from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2$. Then $x_1 x_2$ is affine-equivalent to $x_l \ell(x_1, \dots, x_{l-1}, x_{l+1}, \dots, x_n)$.*

Proof. We have ℓ which is a linear function, which means that

$$\ell(x_1, \dots, x_{l-1}, x_{l+1}, \dots, x_n) = \sum_{i=1, i\neq l}^{n} a_i x_i,$$

where $a_i \in \mathbb{F}_2$. Since ℓ is non-zero, we know that there exists $m \in \{1, \dots, l-1, l+1, \dots, n\}$ such that $a_m \neq 0$. Let us define the $n \times n$ matrix A. We start by defining A as I_n with the following changes:

- the first row is a zero vector except for 1 at the l-th position,
- the second row contains the element a_i at the i-th position for $i \in \{1, \dots, l-1, l+1, \dots, n\}$ and zero at the l-th position,
- the l-th row is a zero vector except for 1 at the first position,
- the m-th row is a zero vector except for 1 at the second position.

Such a defined matrix satisfies $A \in \mathrm{GL}(n, \mathbb{F}_2)$, therefore if we consider the function $x_1 x_2$ as a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$ as $(x_1, x_2, x_3, \dots, x_n) \mapsto x_1 x_2$ we get that

$$x_1 x_2 \circ A(x) = x_l \ell(x_1, \dots, x_{l-1}, x_{l+1}, \dots, x_n).$$

□

The proof of the following proposition is given, in the proof of Proposition 5 in [5, Section 5], with reference only to [8, page 173]. Thus we studied the necessary theory and proved the following proposition.

Proposition 35. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be quadratic APN vectorial Boolean function with maximum linearity. Let $f_1, \dots, f_n : \mathbb{F}_2^n \to \mathbb{F}_2$ be coordinate functions of F. Then there exists quadratic APN vectorial boolean function $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ with maximum linearity such that $G \approx_{\mathrm{EA}} F$ and n-th coordinate function of G is $x_l \ell(x_1, \dots, x_{n-1})$, where ℓ is a linear function.*

Proof. The function F has maximum linearity. This implies that there exist $\alpha \in \mathbb{F}_2^n$ and $\beta \in \mathbb{F}_2^n \setminus \{0\}$ such that

$$\widehat{F_\beta}(\alpha) = 2^{n-1} \qquad \text{or} \qquad \widehat{F_\beta}(\alpha) = -2^{n-1}.$$

Let us denote $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ and $\beta = (\beta_1, \beta_2, \ldots, \beta_n)$ where $\alpha_i, \beta_i \in \mathbb{F}_2$, $i \in \{1, \cdots, n\}$. Since β is fixed, we can define the following set $\mathcal{C} := \{i \mid \beta_i = 1\}$. Let us denote $c_1, \ldots, c_k$ the elements from $\mathcal{C}$, where k is the cardinality of $\mathcal{C}$. Therefore for any $x \in \mathbb{F}_2^n$ we have

$$\begin{aligned}\langle \beta, F(x) \rangle &= \beta_1 f_1(x) \oplus \beta_2 f_2(x) \oplus \cdots \oplus \beta_n f_n(x) \\ &= \beta_{c_1} f_{c_1}(x) \oplus \beta_{c_2} f_{c_2}(x) \oplus \cdots \oplus \beta_{c_k} f_{c_k}(x) \\ &= f_{c_1}(x) \oplus \cdots \oplus f_{c_k}(x)\end{aligned}$$

Let us define $f_{\mathcal{C}}(x) := f_{c_1}(x) \oplus \cdots \oplus f_{c_k}(x)$. From the definition of the Walsh transform we now get

$$\widehat{F_\beta}(\alpha) = \sum_{x \in \mathbb{F}_2^n} (-1)^{\langle \beta, F(x) \rangle \oplus \langle \alpha, x \rangle} = \sum_{x \in \mathbb{F}_2^n} (-1)^{f_{\mathcal{C}}(x) \oplus \langle \alpha, x \rangle} = \widehat{f_{\mathcal{C}}}(\alpha).$$

F is quadratic, thus we have two possible cases for the degree of $f_{\mathcal{C}}(x)$. The first case is that $\deg(f_{\mathcal{C}}) = 2$ and the second case is that $\deg(f_{\mathcal{C}}) \leq 1$.

Let us start with the first case. Since $f_{\mathcal{C}}(x)$ is of degree 2, we know from Theorem 32 that $f_{\mathcal{C}}(x)$ is EA-equivalent to $x_1x_2 \oplus \cdots \oplus x_{2r-1}x_{2r}$, where $r \leq \frac{n}{2}$. We want to find the value of r. Since the absolute value of the Walsh transform is EA-invariant according to Theorem 15, we can write

$$\pm 2^{n-1} = \widehat{F_\beta}(\alpha) = \widehat{f_{\mathcal{C}}}(\alpha) = \sum_{x \in \mathbb{F}_2^n} (-1)^{x_1x_2 \oplus \cdots \oplus x_{2r-1}x_{2r} \oplus \langle \alpha, x \rangle}. \tag{5.2}$$

We can rewrite $\langle \alpha, x \rangle$ as follows

$$\langle \alpha, x \rangle = (\alpha_1 x_1 \oplus \cdots \oplus \alpha_{2r} x_{2r}) \oplus (\alpha_{2r+1} x_{2r+1} \oplus \cdots \oplus \alpha_n x_n).$$

Thus we can rewrite the exponent from the equation 5.2 using the expression from [8, page 173].

$$\begin{aligned}x_1x_2 \oplus &\cdots \oplus x_{2r-1}x_{2r} \oplus (\alpha_1 x_1 \oplus \cdots \oplus \alpha_{2r} x_{2r}) \oplus (\alpha_{2r+1} x_{2r+1} \oplus \cdots \oplus \alpha_n x_n) \\ &= (x_1x_2 \oplus \alpha_1 x_1 \oplus \alpha_2 x_2 \oplus \alpha_1\alpha_2 \oplus \alpha_1\alpha_2) \qquad (5.3) \\ &\quad \oplus (x_3x_4 \oplus \alpha_3 x_3 \oplus \alpha_4 x_4 \oplus \alpha_3\alpha_4 \oplus \alpha_3\alpha_4) \\ &\quad \oplus \cdots \oplus \\ &\quad \oplus (x_{2r-1}x_{2r} \oplus \alpha_{2r-1} x_{2r-1} \oplus \alpha_{2r} x_{2r} \oplus \alpha_{2r-1}\alpha_{2r} \oplus \alpha_{2r-1}\alpha_{2r}) \qquad (5.4) \\ &\quad \oplus \alpha_{2r+1} x_{2r+1} \oplus \alpha_{2r+2} x_{2r+2} \oplus \cdots \oplus \alpha_{n-1} x_{n-1} \oplus \alpha_n x_n\end{aligned}$$

For $i \in \{1, \ldots, r\}$ we can rewrite expressions from 5.3 to 5.4 using

$$(x_{2i-1} \oplus \alpha_{2i})(x_{2i} \oplus \alpha_{2i-1}) = x_{2i-1}x_{2i} \oplus \alpha_{2i-1} x_{2i-1} \oplus \alpha_{2i} x_{2i} \oplus \alpha_{2i-1}\alpha_{2i},$$

therefore the exponent of 5.2 can be expressed as

$$\bigoplus_{i=1}^{r} \left[(x_{2i-1} \oplus \alpha_{2i})(x_{2i} \oplus \alpha_{2i-1}) \oplus \alpha_{2i-1}\alpha_{2i} \right] \oplus \alpha_{2r+1} x_{2r+1} \oplus \cdots \oplus \alpha_n x_n.$$

and thus

$$\begin{aligned}\widehat{f_{\mathcal{C}}}(\alpha) &= \sum_{x\in\mathbb{F}_2^n} (-1)^{\bigoplus_{i=1}^{r}[(x_{2i-1}\oplus\alpha_{2i})(x_{2i}\oplus\alpha_{2i-1})\oplus\alpha_{2i-1}\alpha_{2i}]\oplus\alpha_{2r+1}x_{2r+1}\oplus\cdots\oplus\alpha_n x_n} \qquad (5.5)\\ &= \sum_{x\in\mathbb{F}_2^n} \underbrace{(-1)^{\bigoplus_{i=1}^{r}[(x_{2i-1}\oplus\alpha_{2i})(x_{2i}\oplus\alpha_{2i-1})]}}_{\text{1st part}}\underbrace{(-1)^{\bigoplus_{i=1}^{r}\alpha_{2i-1}\alpha_{2i}}}_{\text{2nd part}}\underbrace{(-1)^{\alpha_{2r+1}x_{2r+1}\oplus\cdots\oplus\alpha_n x_n}}_{\text{3rd part}}\end{aligned}$$

Now we analyse the highlighted parts of the sum. The second part is constant because $\alpha \in \mathbb{F}_2^n$ is fixed, so the second part is equal either to 1 or -1.

In the third part we can see that if we have $\alpha_i \neq 0$ for $i \in \{2r+1, \ldots, n\}$, then for $x \in \mathbb{F}_2^n$, such that $x_i = 0$, the whole exponent has value equal to some δ and for $x \in \mathbb{F}_2^n$, such that $x_i = 1$, the exponent has a value $\delta + 1$. Therefore, such $\alpha_i \neq 0$ causes that the sum in 5.5 is equal to 0. Thus we assume that $\alpha_i = 0$ for $i \in \{2r+1, \ldots, n\}$.

Now for the first part we will show how the values $\alpha_{2i-1}, \alpha_{2i}$ for $i \in \{1, \ldots, r\}$ affect the result of $(x_{2i-1} \oplus \alpha_{2i})(x_{2i} \oplus \alpha_{2i-1})$ since the sum in 5.5 is over all $x \in \mathbb{F}_2^n$. We have four possible cases for the values of the tuple $\alpha_{2i-1}, \alpha_{2i}$ (these are $\{(0,0),(1,0),(0,1),(1,1)\}$) and for each of these cases we have again four possible choices for the values of the tuple x_{2i-1}, x_{2i}. Let us denote $\gamma := (x_{2i-1} \oplus \alpha_{2i})(x_{2i} \oplus \alpha_{2i-1})$.

If we now fix $i \in \{1, \ldots, r\}$, then we can observe that for each fixed value of the tuple $\alpha_{2i-1}, \alpha_{2i}$ we have $\gamma = 1$ for one choice of x_{2i-1}, x_{2i} and $\gamma = 0$ for three choices of x_{2i-1}, x_{2i}. Thus we can see that $(-1)^\gamma = -1$ for one choice of x_{2i-1}, x_{2i} and $(-1)^\gamma = 1$ for three choices of x_{2i-1}, x_{2i}. Therefore, since we consider all $x \in \mathbb{F}_2^n$, we can see that in the the sum 5.5 have impact two choices of x_{2i-1}, x_{2i} such that $\gamma = 0$ and the remaining choice of x_{2i-1}, x_{2i} such that $\gamma = 0$ is annihilated by the choice x_{2i-1}, x_{2i} such that $\gamma = 1$.

Thus for each $i \in \{1, \ldots, r\}$ we have four possible choices for x_{2i-1}, x_{2i} but only two of them affect the sum. So for all $x \in \mathbb{F}_2^n$ we have 2^{n-2r} possible choices of the part $(x_{2r+1}, x_{2r+2}, \ldots, x_n)$ of x. For $(x_1, \ldots, x_{2r})$ only 2^r choices affect the sum, so in total we have $2^{n-2r}2^r = 2^{n-r}$.

All together we have that $\widehat{f_{\mathcal{C}}}(\alpha) = \pm 2^{n-r}$ when $\alpha_i = 0$ for $i \in \{2r+1, \ldots, n\}$ and $\widehat{f_{\mathcal{C}}}(\alpha) = 0$ otherwise. Since we know from the equation 5.2 that $2^{n-1} = \widehat{f_{\mathcal{C}}}(\alpha)$, it must hold that $\widehat{f_{\mathcal{C}}}(\alpha) = \pm 2^{n-r}$, which implies that $r = 1$. Therefore in the case I. we have that $f_{\mathcal{C}}(x)$ is EA-equivalent to x_1x_2.

We assume that $f_{\mathcal{C}}$ is quadratic. This implies that there are $x_lx_m \in f_{\mathcal{C}}(x)$ such that $l, m \in \{1, \ldots, n\}$, $l \neq m$ and $l < m$. Therefore we can rewrite $f_{\mathcal{C}}$ as

$$f_{\mathcal{C}}(x) = x_l\ell(x_1, \ldots, x_{l-1}, x_{l+1} \ldots, x_n) + f_{\mathcal{C}}^r(x)$$

for a linear function ℓ and a quadratic function $f_{\mathcal{C}}^r$ such that $x_lx_m \notin f_{\mathcal{C}}^r(x)$. Since $x_lx_m \in f_{\mathcal{C}}(x)$, we know that $\deg(\ell) = 1$. Therefore we have satisfied the conditions of Lemma 34 for $x_l\ell(x_1, \ldots, x_{l-1}, x_{l+1} \ldots, x_n)$ which implies that $x_l\ell(x_1, \ldots, x_{l-1}, x_{l+1} \ldots, x_n)$ is affine equivalent to x_1x_2. Thus we have

$$f_{\mathcal{C}}(x) \approx_{\mathrm{EA}} x_l\ell(x_1, \ldots, x_{l-1}, x_{l+1} \ldots, x_n).$$

Now we can define a permutation matrix P_ℓ which permutes $x = (x_1, \ldots, x_n)$ into $(x_1, \ldots, x_{l-1}, x_n, x_l, x_{l+1}, x_{l+2}, \ldots, x_{n-2}, x_{n-1})$. Therefore

$$(x_l\ell) \circ P_\ell(x) = (x_l\ell)(x_1, \ldots, x_{l-1}, x_n, x_{l+1}, \ldots, x_{n-1}) = x_n\ell(x_1, \ldots, x_{n-1}).$$

Since $\beta_{c_1} = 1$, we can define the permutation matrix P_F such that P_F is an identity matrix except that on the c_1-th row we have a zero row vector with 1 at the n-th position and the n-th row is β^T. Hence

$$\begin{aligned} P_F \circ F(x) &= P_F \circ (f_1(x), \ldots, f_n(x))^T \\ &= (f_1(x), \ldots, f_{c_{i-1}}(x), f_n(x), f_{c_{i+1}}(x), \ldots, f_{\mathcal{C}}(x))^T. \end{aligned}$$

So we can define

$$(g_1(x), \ldots, g_n(x)) := P_F \circ F \circ P_\ell(x),$$

which satisfies that $g_n(x) = x_n\ell(x_1, \ldots, x_{n-1})$. Thus if we define vectorial Boolean function $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ such that its coordinate functions are g_i, for $i \in \{1, \ldots, n\}$ we get G from the statement of the proposition.

Now we have to prove to prove the statement for the second case, and that is for $\deg(f_{\mathcal{C}}) \leq 1$. This implies that $f_{\mathcal{C}}$ is an affine function, thus $f_{\mathcal{C}}(x) \approx_{\mathrm{EA}} 0$ using

$$0 = I_1 \circ f_{\mathcal{C}} \circ I_1(x) + f_{\mathcal{C}}(x),$$

thus for affine function $B : \mathbb{F}_2^n \to \mathbb{F}_2^n$ such that $B(x) := (0, \ldots, f_{\mathcal{C}} \circ P_\ell(x))$ we can define

$$(g_1(x), \ldots, g_n(x)) := P_F \circ F \circ P_\ell(x) + B(x),$$

where $g_n(x) = 0$.

□

Finally, we can prove the following theorem. This gives us for every vectorial Boolean function $F : \mathbb{F}_2^{n+1} \to \mathbb{F}_2^{n+1}$ an EA-equivalent function T which has a certain form. The theorem is from [5, Section 5]. The proof is described in more detail in this thesis with references to previous lemmas and propositions.

Theorem 36. *Let $F : \mathbb{F}_2^{n+1} \to \mathbb{F}_2^{n+1}$ be a quadratic vectorial Boolean function with linearity 2^n. Then, there exists a function $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ of algebraic degree at most 2 and two linear functions $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$, $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$, $\ell \neq 0$ such that F is EA-equivalent to*

$$\begin{aligned} T : \mathbb{F}_2^n \times \mathbb{F}_2 \quad &\to \quad \mathbb{F}_2^n \times \mathbb{F}_2 \\ \begin{pmatrix} x \\ y \end{pmatrix} \quad &\mapsto \quad \begin{pmatrix} G(x) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} \cdot y. \end{aligned}$$

Proof. Since F is a quadratic vectorial Boolean function with maximum linearity, we can use Proposition 35. Thus we have a quadratic APN vectorial Boolean function $H \approx_{\mathrm{EA}} F$ such that $H(x, y) = (h_1(x, y), \ldots, h_n(x, y), h_{n+1}(x, y))$, where h_i is the i-th coordinate function of H for $i \in \{1, \ldots, n+1\}$ and $h_{n+1}(x, y) = y\ell(x)$ is the $n+1$-th coordinate function of H, where ℓ is linear function. For $t \in \{1, \ldots, n\}$ we denote by

$$h_t(x) = \sum_{u \in \mathbb{F}_2^{n+1}} a_{i,u} \prod_{i=1}^{n+1} x_i^{u_i}$$

ANF of the Boolean function h_t. Now we denote by $e = (e_1, \ldots, e_{n+1}) := (0, \ldots, 0, 1)$ the element with n zeros and one on the $n+1$-th position. By

0^{n+1} we mean the zero element of $\mathbb{F}_2^{n+1}$. Let us choose any $t \in \{1, \ldots, n\}$. We now evaluate h_t on e_{n+1} and 0^{n+1}. We start with e_{n+1}. Since H is quadratic, then h_t is at most quadratic, thus without loss of generality let h_t be quadratic. Hence

$$h_t(e) = \sum_{\mathrm{wt_H}(u) \leq 2,\ u \in \mathbb{F}_2^{n+1}} a_{i,u} \prod_{i=1}^{n+1} e_i^{u_i}.$$

For each u such that $\mathrm{wt_H}(u) = 2$ we have indices $j, k \in \{1, \ldots, n+1\}$, $j \neq k$ such that $u_j = u_k = 1$. Thus in the summand for such u we have a product of e_j and e_k, where at least one of e_j and e_k is equal to zero. Thus the product is equal to zero which implies that the whole summand is equal to zero. Therefore we can omit $u \in \mathbb{F}_2^{n+1}$ such that $\mathrm{wt_H}(u) = 2$, thus we can rewrite h_t as follows

$$h_t(e) = \sum_{\mathrm{wt_H}(u) \leq 1,\ u \in \mathbb{F}_2^{n+1}} a_{i,u} \prod_{i=1}^{n+1} e_i^{u_i}.$$

If we have u such that $\mathrm{wt_H}(u) = 1$, then we have $j \in \{1, \ldots, n+1\}$ for which $u_j = 1$. Since $0^1 = 0$ we are only interested in those u where $u_{n+1} = 1$, therefore we are only interested in one element u and that is $u = e$.

There is only one element for which $\mathrm{wt_H}(u) = 0$, and that is 0^{n+1}. Therefore we can rewrite h_t again and so we have

$$\begin{aligned} h_t(e) &= \sum_{\mathrm{wt_H}(u)=0,\ u \in \mathbb{F}_2^{n+1}} a_{i,u} \prod_{i=1}^{n+1} e_i^{u_i} + \sum_{\mathrm{wt_H}(u)=1,\ u \in \mathbb{F}_2^{n+1}} a_{i,u} \prod_{i=1}^{n+1} e_i^{u_i} \\ &= a_{i,e} + a_{i,0^{n+1}}. \end{aligned}$$

Using the same argument we get the $h_t(0^{n+1}) = a_{i,0^{n+1}}$. Now we define the affine function $B : \mathbb{F}_2^{n+1} \to \mathbb{F}_2^{n+1}$ as follows

$$B(x,y) = \begin{pmatrix} 0 & 0 & \cdots & 0 & a_{1,e} \\ 0 & 0 & \cdots & 0 & a_{2,e} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & a_{n,e} \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ y \end{pmatrix} + \begin{pmatrix} a_{1,0^{n+1}} \\ a_{2,0^{n+1}} \\ \vdots \\ a_{n,0^{n+1}} \\ 0 \end{pmatrix},$$

Since B is an affine function we can define the EA-equivalent function T to F as follows

$$T = I_{n+1} H I_{n+1} + B.$$

Such a function satisfies

$$T(e) = H(e) + B(e) = 0^{n+1} \quad \text{and} \quad T(0^{n+1}) = H(0^{n+1}) + B(0^{n+1}) = 0^{n+1},$$

since also $h_{n+1}(e) = 0$ and $h_{n+1}(0^{n+1}) = 0$.

The vectorial Boolean function T can be expressed by its coordinate functions $t_1, \ldots, t_{n+1}$, where $t_j : \mathbb{F}_2^{n+1} \to \mathbb{F}_2$ for $j \in \{1, \ldots, n+1\}$ and $t_{n+1}(x,y) = h_{n+1}(x,y)$. Using Lemma 33 gives us that the function $x \mapsto t_i(x,0) + t_i(x,1)$ is of degree at most 1 for any $i \in \{1, \ldots, n\}$. This implies that there exists an affine function $l_i : \mathbb{F}_2^n \to \mathbb{F}_2$ such that

$$l_i(x) = t_i(x,0) + t_i(x,1) \implies t_i(x,1) = t_i(x,0) + l_i(x).$$

Since we know that for our choice of T it holds that $t_i(0^{n+1}) = t_i(e) = 0$, we have that

$$l_i(\underbrace{0,\dots,0}_{n \text{ times}}) = t_i(e) + t_i(0^{n+1}) = 0,$$

which implies that l_i is linear, since the constant from the definition of the affine function is zero. Any $t_i(x,y)$ can be expressed in such a way that

$$t_i\begin{pmatrix}x\\y\end{pmatrix} = t_i\begin{pmatrix}x\\0\end{pmatrix}(y+1) + t_i\begin{pmatrix}x\\1\end{pmatrix}y.$$

Using the linear function l_i, we can rewrite it in a way that as

$$\begin{aligned}t_i\begin{pmatrix}x\\y\end{pmatrix} &= t_i\begin{pmatrix}x\\0\end{pmatrix}(y+1) + t_i\begin{pmatrix}x\\1\end{pmatrix}y\\ &= t_i\begin{pmatrix}x\\0\end{pmatrix}y + t_i\begin{pmatrix}x\\0\end{pmatrix} + t_i\begin{pmatrix}x\\0\end{pmatrix}y + l_i(x)y\\ &= t_i\begin{pmatrix}x\\0\end{pmatrix} + l_i(x)y.\end{aligned}$$

Now let us define the functions $g_i(x) := t_i(x,0)$ for $i \in \{1,\dots,n\}$ and the vectorial Boolean function $G(x) := (g_1(x),\dots,g_n(x))$. Let also L be a linear function such that the i-th row is equal to l_i for $i \in \{1,\dots,n\}$, so $L(x) = (l_1(x),\dots,l_n(x))^T$. We know that $h_{n+1} = \ell(x)y$, therefore

$$\begin{aligned}T(x,y) &= \begin{pmatrix}t_1(x,y)\\\vdots\\t_n(x,y)\\t_{n+1}(x,y)\end{pmatrix} = \begin{pmatrix}t_1(x,0)+l_1(x)y\\\vdots\\t_n(x,0)+l_n(x)y\\\ell(x)y\end{pmatrix} = \begin{pmatrix}g_1(x)\\\vdots\\g_n(x)\\0\end{pmatrix} + \begin{pmatrix}l_1(x)\\\vdots\\l_n(x)\\\ell(x)\end{pmatrix}y\\ &= \begin{pmatrix}G(x)\\0\end{pmatrix} + \begin{pmatrix}L(x)\\\ell(x)\end{pmatrix}y.\end{aligned}$$

□

Now we want to show, that the G in the Theorem 36 can be chosen up to EA-equivalence. Thus we prove the following proposition, which is given as a remark in [5, Section 5] without proof.

Proposition 37. *Let $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$, $\ell \neq 0$ be linear mappings. Let G and G' be two vectorial Boolean functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$ such that $G \approx_{\mathrm{EA}} G'$. Then there exist linear mappings $L' : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $\ell' : \mathbb{F}_2^n \to \mathbb{F}_2$, $\ell' \neq 0$ such that the two functions*

$$\begin{pmatrix}x\\y\end{pmatrix} \mapsto \begin{pmatrix}G(x)\\0\end{pmatrix} + \begin{pmatrix}L(x)\\\ell(x)\end{pmatrix}\cdot y, \qquad \begin{pmatrix}x\\y\end{pmatrix} \mapsto \begin{pmatrix}G'(x)\\0\end{pmatrix} + \begin{pmatrix}L'(x)\\\ell'(x)\end{pmatrix}\cdot y$$

are EA-equivalent.

Proof. We know that $G \approx_{\mathrm{EA}} G'$, therefore from the definition of EA-equivalence we have the existence of A_2, A_1 and B such that $G(x) = A_2 \circ G' \circ A_1(x) + B(x)$. Let T be the function

$$\begin{aligned} T : \mathbb{F}_2^n \times \mathbb{F}_2 &\to \mathbb{F}_2^n \times \mathbb{F}_2 \\ \begin{pmatrix} x \\ y \end{pmatrix} &\mapsto \begin{pmatrix} G(x) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} \cdot y. \end{aligned}$$

and T' be the function

$$\begin{aligned} T' : \mathbb{F}_2^n \times \mathbb{F}_2 &\to \mathbb{F}_2^n \times \mathbb{F}_2 \\ \begin{pmatrix} x \\ y \end{pmatrix} &\mapsto \begin{pmatrix} G'(x) \\ 0 \end{pmatrix} + \begin{pmatrix} L'(x) \\ \ell'(x) \end{pmatrix} \cdot y. \end{aligned}$$

We want to show, that $T \approx_{\mathrm{EA}} T'$. Let us first use the fact, that $G \approx_{\mathrm{EA}} G'$.

$$\begin{aligned} T &= \begin{pmatrix} G(x) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} \cdot y \\ &= \begin{pmatrix} A_2 \circ G' \circ A_1(x) + B(x) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} \cdot y \\ &= \begin{pmatrix} A_2 \circ G' \circ A_1(x) + L(x)y \\ \ell(x)y \end{pmatrix} + \begin{pmatrix} B(x) \\ 0 \end{pmatrix} \end{aligned}$$

Since A_1 and A_2 are bijections, we can write

$$= \begin{pmatrix} A_2 \circ G' \circ A_1(x) + (A_2 \circ A_2^{-1} \circ L \circ A_1^{-1} \circ A_1(x))y \\ (\ell \circ A_1^{-1} \circ A_1(x))y \end{pmatrix} + \begin{pmatrix} B(x) \\ 0 \end{pmatrix}.$$

Now we define $L' := A_2^{-1} \circ L \circ A_1^{-1}$ and $\ell' := \ell \circ A_1^{-1}$. Therefore we get

$$= \begin{pmatrix} A_2 \circ (G' + L'y) \circ A_1(x) \\ (\ell' \circ A_1(x))y \end{pmatrix} + \begin{pmatrix} B(x) \\ 0 \end{pmatrix}.$$

We can also define A_2' and A_1' as

$$\begin{aligned} A_2' : \mathbb{F}_2^n \times \mathbb{F}_2 &\to \mathbb{F}_2^n \times \mathbb{F}_2 \\ \begin{pmatrix} x \\ y \end{pmatrix} &\mapsto \begin{pmatrix} A_2(x) \\ y \end{pmatrix} \end{aligned} \qquad \begin{aligned} A_1' : \mathbb{F}_2^n \times \mathbb{F}_2 &\to \mathbb{F}_2^n \times \mathbb{F}_2 \\ \begin{pmatrix} x \\ y \end{pmatrix} &\mapsto \begin{pmatrix} A_2(x) \\ y \end{pmatrix} \end{aligned}$$

and B' as

$$\begin{aligned} B' : \mathbb{F}_2^n \times \mathbb{F}_2 &\to \mathbb{F}_2^n \times \mathbb{F}_2 \\ \begin{pmatrix} x \\ y \end{pmatrix} &\mapsto \begin{pmatrix} B(x) \\ 0 \end{pmatrix}. \end{aligned}$$

Therefore, both A_2' and A_1' are still affine bijections, and B' is still affine function. Therefore, we can write

$$= A_2' \circ \left(\begin{pmatrix} G' \\ 0 \end{pmatrix} + \begin{pmatrix} L' \\ \ell' \end{pmatrix} y \right) \circ A_1' \begin{pmatrix} x \\ y \end{pmatrix} + B' \begin{pmatrix} x \\ y \end{pmatrix}.$$

This is the definition of $T \approx_{\mathrm{EA}} T'$, so we have proved the proposition.

$\square$

Since we are interested in APN functions, we state the following theorem which gives us conditions under which the function T from Theorem 36 is APN. The theorem is given as Theorem 1 in [5, Section 5]. The proof is described in more detail in this thesis.

Theorem 38. *Let $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic vectorial Boolean function. Let $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$, $\ell \neq 0$ be two linear functions. Then*

$$\begin{aligned} T : \mathbb{F}_2^n \times \mathbb{F}_2 &\to \mathbb{F}_2^n \times \mathbb{F}_2 \\ \begin{pmatrix} x \\ y \end{pmatrix} &\mapsto \begin{pmatrix} G(x) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} y \end{aligned}$$

is APN if and only if the following two assertions hold:

1. *G is APN*
2. *$\langle \pi_G(\alpha), L(\alpha) \rangle = 1$ for all $\alpha \in \mathbb{F}_2^n \setminus \{0\}$ with $\ell(\alpha) = 0$.*

Proof. Recall, that if T is an APN function, then it means from the definition it follows that $\forall a \in \mathbb{F}_2^{n+1}$, $a \neq 0$ and $\forall b \in \mathbb{F}_2^{n+1}$ the equation for any $z \in \mathbb{F}_2^{n+1}$

$$T(z) + T(z + a) = b$$

has at most two solutions. This can be written as $\forall \alpha, \gamma \in \mathbb{F}_2^n$ and $\forall \beta, \delta \in \mathbb{F}_2$ ($(\alpha, \beta) \neq (0, 0)$), the equation

$$T \begin{pmatrix} x \\ y \end{pmatrix} + T \begin{pmatrix} x + \alpha \\ y + \beta \end{pmatrix} = \begin{pmatrix} \gamma \\ \delta \end{pmatrix}$$

has at most two solutions $(x, y) \in \mathbb{F}_2^n \times \mathbb{F}_2$. Using the definition of T we get

$$\begin{aligned} T \begin{pmatrix} x \\ y \end{pmatrix} + T \begin{pmatrix} x + \alpha \\ y + \beta \end{pmatrix} &= \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \\ \begin{pmatrix} G(x) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} y + \begin{pmatrix} G(x+\alpha) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x+\alpha) \\ \ell(x+\alpha) \end{pmatrix} (y + \beta) &= \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \\ \begin{pmatrix} G(x) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} y + \begin{pmatrix} G(x+\alpha) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x) + L(\alpha) \\ \ell(x) + \ell(\alpha) \end{pmatrix} (y + \beta) &= \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \\ \begin{pmatrix} G(x) + G(x+\alpha) \\ 0 \end{pmatrix} + \begin{pmatrix} L(\alpha) \\ \ell(\alpha) \end{pmatrix} y + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} \beta + \begin{pmatrix} L(\alpha) \\ \ell(\alpha) \end{pmatrix} \beta &= \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \end{aligned} \tag{5.6}$$

Since $(\alpha, \beta) \neq (0, 0)$ and $\beta \in \mathbb{F}_2$, we can divide the proof into two separable cases. In the first case we assume that $\beta = 0$, which implies that α is not a zero vector. In the second case we will assume that $\beta = 1$, so α can be a zero vector. So let us start with the first case. $\beta = 0$ and fix some arbitrary α. For our fixed α we recall that the mapping $B_\alpha : \mathbb{F}_2^n \to \mathbb{F}_2^n$, $x \mapsto G(x) + G(x + \alpha) + G(0) + G(\alpha)$ is linear due to Lemma 29.

Now we divide the proof again into two cases. First we assume that $y = 0$. Then we have $\beta = 0, y = 0$ and α is a non-zero vector from $\mathbb{F}_2^n$. We assume that T is an APN function.

$$\begin{aligned}
T\begin{pmatrix} x \\ y \end{pmatrix} + T\begin{pmatrix} x+\alpha \\ y+\beta \end{pmatrix} &= \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \\
T\begin{pmatrix} x \\ 0 \end{pmatrix} + T\begin{pmatrix} x+\alpha \\ 0+0 \end{pmatrix} &= \begin{pmatrix} G(x)+G(x+\alpha) \\ 0 \end{pmatrix} + \begin{pmatrix} L(\alpha) \\ \ell(\alpha) \end{pmatrix} 0 + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} 0 + \begin{pmatrix} L(\alpha) \\ \ell(\alpha) \end{pmatrix} 0 \\
&= \begin{pmatrix} G(x)+G(x+\alpha) \\ 0 \end{pmatrix}
\end{aligned}$$

Therefore

$$T\begin{pmatrix} x \\ 0 \end{pmatrix} + T\begin{pmatrix} x+\alpha \\ 0 \end{pmatrix} = \begin{pmatrix} G(x)+G(x+\alpha) \\ 0 \end{pmatrix} = \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \tag{5.7}$$

This implies that $\delta = 0$. For the sake of contradiction, let us assume that G is not APN. Then the equation $G(x) + G(x+\alpha) = \gamma$ has at least three solutions. Without loss of generality, let us assume that the number of solutions is three. Denote them x_1, x_2, x_3. Thus this three solutions satisfy

$$T\begin{pmatrix} x_i \\ 0 \end{pmatrix} + T\begin{pmatrix} x_i+\alpha \\ 0 \end{pmatrix} = \begin{pmatrix} G(x_i)+G(x_i+\alpha) \\ 0 \end{pmatrix} = \begin{pmatrix} \gamma \\ \delta \end{pmatrix},$$

where $i \in \{1, 2, 3\}$. But this is in contradiction with T being an APN function. The second property that $\langle \pi_G(\alpha), L(\alpha)\rangle = 1$ for all $\alpha \in \mathbb{F}_2^n \setminus \{0\}$ with $\ell(\alpha) = 0$ is satisfied, since in this case we have shown, that the equation 5.7 is independent on $L(\alpha)$ and $\ell(\alpha)$.

Now let us assume that the two conditions from the statement are satisfied. We want to show that T is then an APN function. The proof is straightforward, since we assume that G is an APN function and we have the following equation

$$T\begin{pmatrix} x \\ 0 \end{pmatrix} + T\begin{pmatrix} x+\alpha \\ 0 \end{pmatrix} = \begin{pmatrix} G(x)+G(x+\alpha) \\ 0 \end{pmatrix} = \begin{pmatrix} \gamma \\ \delta \end{pmatrix}.$$

Thus T is an APN function.

Let us further assume that since T is an APN function, that $(s_1, 0)$ and $(s_1', 0)$ are the only solutions for some fixed α, γ and $\beta = \delta = 0$. Now suppose $\beta = 0, y = 1$ and α is a non-zero vector from $\mathbb{F}_2^n$.

$$\begin{aligned}
T\begin{pmatrix} x \\ 1 \end{pmatrix} + T\begin{pmatrix} x+\alpha \\ 1 \end{pmatrix} &= \begin{pmatrix} G(x)+G(x+\alpha) \\ 0 \end{pmatrix} + \begin{pmatrix} L(\alpha) \\ \ell(\alpha) \end{pmatrix} 1 + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} 0 + \begin{pmatrix} L(\alpha) \\ \ell(\alpha) \end{pmatrix} 0 \\
&= \begin{pmatrix} G(x)+G(x+\alpha)+L(\alpha) \\ \ell(\alpha) \end{pmatrix} \\
&= \begin{pmatrix} \gamma \\ \delta \end{pmatrix}
\end{aligned}$$

T is an APN function, therefore the equation above have at most two solutions, denoted for example by x_1, x_2. Therefore $G(x) + G(x+\alpha) + L(\alpha) = \gamma$ must be satisfied at most for these two solutions x_1, x_2 since $\ell(\alpha)$ is a constant for a fixed choice of α. $L(\alpha)$ is also a constant, so from $G(x) + G(x+\alpha) + L(\alpha) = \gamma$ we know that G must be an APN function.

Now we want to prove the second property. From the equation of B_α we know that

$$G(x) + G(x+\alpha) = B_\alpha(x) + G(\alpha) + G(0).$$

Therefore

$$T\begin{pmatrix} x \\ 1 \end{pmatrix} + T\begin{pmatrix} x+\alpha \\ 1 \end{pmatrix} = \begin{pmatrix} G(x) + G(x+\alpha) + L(\alpha) \\ \ell(\alpha) \end{pmatrix} = \begin{pmatrix} \gamma \\ \delta \end{pmatrix}$$

$$\begin{aligned}
&\iff \begin{pmatrix} \gamma + G(x) + G(x+\alpha) + L(\alpha) \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ \delta \end{pmatrix} \\
&\iff \begin{pmatrix} \gamma + G(\alpha) + G(0) + L(\alpha) \\ 0 \end{pmatrix} = \begin{pmatrix} B_\alpha(x) \\ \delta \end{pmatrix} \\
&\iff B_\alpha(x) = \gamma + G(\alpha) + G(0) + L(\alpha)
\end{aligned}$$

Now let us assume that $\gamma + G(\alpha) + G(0) \in \mathrm{Im}(B_\alpha)$ and $\ell(\alpha) = 0$. This means that there exists some s_2 such that $B_\alpha(s_2) = \gamma + G(\alpha) + G(0)$. Moreover

$$B_\alpha(s_2) = \gamma + G(\alpha) + G(0) \ \& \ B_\alpha(s_2) = G(s_2) + G(s_2+\alpha) + G(0) + G(\alpha)$$

which implies that $G(s_2) + G(s_2+\alpha) = \gamma$. Since G is an APN function, we can assume that there is also s_2' which satisfies the same as s_2. Also, since

$$B_\alpha(s_2) = \gamma + G(\alpha) + G(0) \ \& \ B_\alpha(s_2) = \gamma + G(\alpha) + G(0) + L(\alpha),$$

it is equivalent to $L(s_2)$ being a zero vector. Overall, since $\ell(\alpha) = 0$, $L(\alpha)$ is a zero vector, then

$$T\begin{pmatrix} s_2 \\ 1 \end{pmatrix} + T\begin{pmatrix} s_2+\alpha \\ 1 \end{pmatrix} = \begin{pmatrix} G(s_2) + G(s_2+\alpha) \\ 0 \end{pmatrix} = \begin{pmatrix} \gamma \\ 0 \end{pmatrix}.$$

Therefore $(s_2, 1)$ and $(s_2', 1)$ are solutions of the equation for some fixed α, γ and $\beta = \delta = 0$. But $(s_2, 0)$ and $(s_2', 0)$ are also solutions of the equation 5.7 which implies, that we have four solutions for some fixed α, γ and $\beta = \delta = 0$. This is a contradiction.

Therefore $\gamma + G(\alpha) + G(0) \notin \mathrm{Im}(B_\alpha)$ whenever $\ell(\alpha) = 0$ and since $B_\alpha(x) = \gamma + G(\alpha) + G(0) + L(\alpha)$ it implies that $L(\alpha) \notin \mathrm{Im}(B_\alpha)$. From the definition of B_α and ortho-derivative function $\pi_G(\alpha)$ we know that $\mathrm{Im}(B_\alpha) = \{x \in \mathbb{F}_2^n \mid \langle \pi_G(\alpha), x \rangle = 0\}$. Thus $L(\alpha) \notin \mathrm{Im}(B_\alpha) \iff \langle \pi_G(\alpha), L(\alpha) \rangle = 1$.

Now for the other implication. So let us assume, that the two conditions from the statement are true. Since we assume that $\langle \pi_G, L(\alpha) \rangle = 1$ for α non-zero vector from $\mathbb{F}_2^n$ with $\ell(\alpha) = 0$, then $L(\alpha)$ is a non-zero vector. This means, that every solution of

$$T\begin{pmatrix} x \\ 1 \end{pmatrix} + T\begin{pmatrix} x+\alpha \\ 1 \end{pmatrix} = \begin{pmatrix} G(x) + G(x+\alpha) + L(\alpha) \\ \ell(\alpha) \end{pmatrix} = \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \tag{5.8}$$

for some fixed α, γ and $\beta = \delta = 0$ cannot be a solution of the equation 5.7. We can rearrange the equation 5.8 as follows

$$T\begin{pmatrix} x \\ 1 \end{pmatrix} + T\begin{pmatrix} x+\alpha \\ 1 \end{pmatrix} = \begin{pmatrix} G(x)+G(x+\alpha) \\ 0 \end{pmatrix} = \begin{pmatrix} \gamma + L(\alpha) \\ \delta \end{pmatrix}.$$

Since $L(\alpha)$ is a constant for some fixed α, and we assume that G is an APN function, this implies that T is also an APN function.

Now we move to the case $\beta = 1$. Thus $\alpha \in \mathbb{F}_2^n$. Suppose T is an APN function. We want to show that the two conditions from the statement hold. From the equation 5.6 we get for $y = 0$

$$\begin{aligned} T\begin{pmatrix} x \\ 0 \end{pmatrix} + T\begin{pmatrix} x+\alpha \\ 1 \end{pmatrix} &= \begin{pmatrix} G(x)+G(x+\alpha) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} + \begin{pmatrix} L(\alpha) \\ \ell(\alpha) \end{pmatrix} \\ &= \begin{pmatrix} G(x)+G(x+\alpha)+L(x+\alpha) \\ \ell(x+\alpha) \end{pmatrix} = \begin{pmatrix} \gamma \\ \delta \end{pmatrix} \end{aligned} \tag{5.9}$$

and for $y = 1$

$$\begin{aligned} T\begin{pmatrix} x \\ 1 \end{pmatrix} + T\begin{pmatrix} x+\alpha \\ 0 \end{pmatrix} &= \begin{pmatrix} G(x)+G(x+\alpha) \\ 0 \end{pmatrix} + \begin{pmatrix} L(\alpha) \\ \ell(\alpha) \end{pmatrix} + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} + \begin{pmatrix} L(\alpha) \\ \ell(\alpha) \end{pmatrix} \\ &= \begin{pmatrix} G(x)+G(x+\alpha)+L(x) \\ \ell(x) \end{pmatrix} = \begin{pmatrix} \gamma \\ \delta \end{pmatrix}. \end{aligned} \tag{5.10}$$

Let us assume for a contradiction that G is not an APN function. Then for some value $\epsilon \in \mathbb{F}_2^n$ we have that x_1, x_2 and x_3 are solutions as follows

$$\epsilon = G(x_1) + G(x_1+\alpha) = G(x_2) + G(x_2+\alpha) = G(x_3) + G(x_3+\alpha).$$

From the equation 5.9 we know that $G(x) + G(x+\alpha) = \gamma + L(x) + L(\alpha)$. We can assume, that $\epsilon = \gamma + L(x_1) + L(\alpha)$. Therefore

$$\epsilon + L(x_1+\alpha) = \gamma = \epsilon + L(x_1) + L(\alpha)$$

and also

$$\begin{aligned} \epsilon + L(x_2+\alpha) &= \gamma = \epsilon + L(x_2) + L(\alpha) \\ \epsilon + L(x_3+\alpha) &= \gamma = \epsilon + L(x_3) + L(\alpha) \end{aligned}$$

This implies that $L(x_1) = L(x_2) = L(x_3)$. Thus for $i \in \{1,2,3\}$ we have

$$T\begin{pmatrix} x_i \\ 0 \end{pmatrix} + T\begin{pmatrix} x_i+\alpha \\ 1 \end{pmatrix} = \begin{pmatrix} \gamma \\ \delta \end{pmatrix}$$

but this contradicts the fact that T is an APN function. Thus G is APN.

Now the second condition of the statement. It is easy to see that $x + \alpha$ is a solution of 5.9 if and only if x is a solution of 5.10. Therefore, the equation 5.6 has at most two solutions if and only if the equation 5.10 has at most one solution. This is the case if and only if the following mapping is injective.

$$\begin{aligned} H_\alpha : \mathbb{F}_2^n &\to \mathbb{F}_2^n \times \mathbb{F}_2 \\ x &\mapsto \begin{pmatrix} G(x)+G(x+\alpha)+L(x) \\ \ell(x) \end{pmatrix} \end{aligned}$$

Now we use a little trick. Let us pick an arbitrary element $x \in \mathbb{F}_2^n$ and a non-zero element $\omega \in \mathbb{F}_2^n$. Let us consider

$$H_\alpha(x) + H_\alpha(x+\omega) = \begin{pmatrix} G(x) + G(x+\alpha) + G(x+\omega) + G(x+\omega+\alpha) + L(\omega) \\ \ell(\omega) \end{pmatrix}.$$

We know that G is of algebraic degree at most 2, thus the mapping

$$x \mapsto G(x) + G(x+\alpha) + G(x+\omega) + G(x+\omega+\alpha)$$

is constant because of the following. Let us assume that $x = (x_1, \dots, x_n)$, $\alpha = (\alpha_1, \dots, \alpha_n)$, $\omega = (\omega_1, \dots, \omega_n)$. If G contains a quadratic element as a result of multiplication of the i-th and j-th coordination, then $G(x) + G(x+\alpha) + G(x+\omega) + G(x+\omega+\alpha)$ will only remain constant since

$$x_i x_j + (x_i + \alpha_i)(x_j + \alpha_j) + (x_i + \omega_i)(x_j + \omega_j) + (x_i + \omega_i + \alpha_i)(x_j + \omega_j + \alpha_j)$$

is equal to $\alpha_i \omega_j + \alpha_j \omega_i$. If it contains a linear element, then

$$x_i + (x_i + \alpha_i) + (x_i + \omega_i) + (x_i + \omega_i + \alpha_i) = 0.$$

Therefore,

$$H_\alpha(x) + H_\alpha(x+\omega) = H_\alpha(0) + H_\alpha(\omega) = \begin{pmatrix} B_\omega(\alpha) + L(\omega) \\ \ell(\omega) \end{pmatrix}.$$

To prove, that H_α is injective, we look at the kernel of the mapping, that is, when the right-hand side is equal to zero. If the right-hand side is equal to 0, this implies that $H(0)$ and $H(\omega)$ map onto the same element. The right-hand side is equal to the zero vector if and only if $L(\omega) \in \operatorname{Im}(B_\omega)$ and $\ell(\omega) = 0$. Thus, for the mapping H_α to be injective, we need to $L(\omega) \notin \operatorname{Im}(B_\omega)$ for all ω non-zero vectors with $\ell(\omega) = 0$.

□

For the following, we need to give two definitions and a lemma. Definition 42 and Definition 43 are given in [5, Section 5]. Lemma 39 is based on the comment from [5, Section 5] after the proof of Theorem 1.

Definition 42. *Let $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN vectorial Boolean function. Let $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be linear and $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$, $\ell \neq 0$ such that*

$$\begin{array}{rcl} T : \mathbb{F}_2^n \times \mathbb{F}_2 & \to & \mathbb{F}_2^n \times \mathbb{F}_2 \\ \begin{pmatrix} x \\ y \end{pmatrix} & \mapsto & \begin{pmatrix} G(x) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix} \cdot y \end{array}$$

is APN. If T is APN, we say that the tuple $(G, 0, L, \ell)$ yields *an APN function T and we say that T is an APN* extension of G in standard form*.*

Lemma 39. *Let G be a quadratic vectorial Boolean function. Let $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$, $\ell \neq 0$ be linear functions such that the tuple $(G, 0, L, \ell)$ yields an APN function T. Then L is of rank n or $n-1$.*

Proof. Consider a mapping

$$\begin{aligned} H_\alpha : \mathbb{F}_2^n &\to \mathbb{F}_2^n \times \mathbb{F}_2 \\ x &\mapsto \begin{pmatrix} G(x) + G(x+\alpha) + L(x) \\ \ell(x) \end{pmatrix}. \end{aligned}$$

We know from the end of the proof of Theorem 38 that for a non-zero element $\omega \in \mathbb{F}_2^n$ which satisfies the condition of the second statement of Theorem 38, H_α is injective. Suppose α is a zero vector. The mapping

$$\begin{aligned} H_0 : \mathbb{F}_2^n &\to \mathbb{F}_2^n \times \mathbb{F}_2 \\ x &\mapsto \begin{pmatrix} L(x) \\ \ell(x) \end{pmatrix}. \end{aligned}$$

is still injective, which means, that the matrix $\begin{pmatrix} L \\ \ell \end{pmatrix}$ must be of a rank n, since the matrix is of a dimension $(n+1) \times n$. Since ℓ is a row vector, the rank of L can only be n or $n-1$.

□

Definition 43. *Let $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN vectorial Boolean function and π_G be ortho-derivative of G. Let $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$ be linear mapping such that $\ell \neq 0$. We define the following set*

$$\Gamma_{G,\ell} := \{L \in \mathcal{L}(\mathbb{F}_2^n, \mathbb{F}_2^n) \mid \langle \pi_G(x), L(x) \rangle = 1 \textit{ for all } x \in \mathbb{F}_2^n \setminus \{0\} \textit{ with } \ell(x) = 0\}.$$

From Theorem 38 we have conditions under which the function T is APN. Now we are interested whether some choice of linear mappings L and ℓ from Theorem 38 gives us an EA-equivalent function to T. This is in the scope of the following proposition, which is Proposition 6 from [5, Section 5]. The proof is described in more detail in this thesis.

Proposition 40. *Let $n \in \mathbb{N}$, $n \geq 3$. Let $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic mapping, $L : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be linear and $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$, $l \neq 0$ be linear such that $(G, 0, L, \ell)$ yields an APN function T. For all $\mu, \nu \in \mathbb{F}_2^n$, the tuple $(G, 0, L + B_\mu + \ell\nu, \ell)$ yields an APN function EA-equivalent to T, where $B_\mu : \mathbb{F}_2^n \to \mathbb{F}_2^n$, $x \mapsto G(x) + G(x+\mu) + G(\mu) + G(0)$.*

The functions $L + B_\mu + \ell\nu$ are pairwise distinct for $\mu, \nu \in \mathbb{F}_2^n$. Moreover, for every $\mu \in \mathbb{F}_2^n$, we have

$$\begin{aligned} |\{\nu \in \mathbb{F}_2^n \mid \mathrm{Rank}(L + B_\mu + \ell\nu) = n\}| &= 2^{n-1}, \\ |\{\nu \in \mathbb{F}_2^n \mid \mathrm{Rank}(L + B_\mu + \ell\nu) = n-1\}| &= 2^{n-1}. \end{aligned}$$

Proof. For an element $c \in \mathbb{F}_2^n$, let us consider the linear involution

$$\begin{aligned} M_c : \mathbb{F}_2^n \times \mathbb{F}_2 &\to \mathbb{F}_2^n \times \mathbb{F}_2 \\ \begin{pmatrix} x \\ y \end{pmatrix} &\mapsto \begin{pmatrix} x + cy \\ y \end{pmatrix}. \end{aligned}$$

For $\mu \in \mathbb{F}_2^n$, let us consider the function

$$\begin{array}{rcl} T'_\mu : \mathbb{F}_2^n \times \mathbb{F}_2 & \to & \mathbb{F}_2^n \times \mathbb{F}_2 \\ \begin{pmatrix} x \\ y \end{pmatrix} & \mapsto & TM_\mu \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} L(\mu) + G(\mu) + G(0) \\ \ell(\mu) \end{pmatrix} y. \end{array}$$

The function M_μ is a linear involution, thus it is an affine permutation. The function

$$\left(\begin{pmatrix} L(\mu) + G(\mu) + G(0) \\ \ell(\mu) \end{pmatrix} y \right) \begin{pmatrix} x \\ y \end{pmatrix}$$

is linear, thus it is an affine function. This means that $T'_\mu \approx_{\mathrm{EA}} T$. We can rewrite T'_μ as follows

$$\begin{aligned} T'_\mu \begin{pmatrix} x \\ y \end{pmatrix} &= TM_\mu \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} L(\mu) + G(\mu) + G(0) \\ \ell(\mu) \end{pmatrix} y \\ &= T \begin{pmatrix} x + \mu y \\ y \end{pmatrix} + \begin{pmatrix} L(\mu) + G(\mu) + G(0) \\ \ell(\mu) \end{pmatrix} y \\ &= \begin{pmatrix} G(x + \mu y) \\ 0 \end{pmatrix} + \begin{pmatrix} L(x + \mu y) \\ \ell(x + \mu y) \end{pmatrix} y + \begin{pmatrix} L(\mu) + G(\mu) + G(0) \\ \ell(\mu) \end{pmatrix} y \end{aligned}$$

Since $y \in \mathbb{F}_2$, we can rewrite the first two summands and get the following

$$\begin{aligned} &= \begin{pmatrix} G(x + \mu y) \\ 0 \end{pmatrix} (y+1) + \begin{pmatrix} G(x + \mu y) + L(x + \mu y) \\ \ell(x + \mu y) \end{pmatrix} y \\ &+ \begin{pmatrix} L(\mu) + G(\mu) + G(0) \\ \ell(\mu) \end{pmatrix} y \end{aligned}$$

The first summand is only relevant if $y = 0$ and the second summand is only relevant if $y = 1$. Therefore

$$\begin{aligned} &= \begin{pmatrix} G(x) \\ 0 \end{pmatrix} (y+1) + \begin{pmatrix} G(x + \mu) + L(x + \mu) \\ \ell(x + \mu) \end{pmatrix} y + \begin{pmatrix} L(\mu) + G(\mu) + G(0) \\ \ell(\mu) \end{pmatrix} y \\ &= \begin{pmatrix} G(x) \\ 0 \end{pmatrix} (y+1) + \begin{pmatrix} G(x + \mu) + L(x) + L(\mu) + L(\mu) + G(\mu) + G(0) \\ \ell(x) + \ell(\mu) + \ell(\mu) \end{pmatrix} y \\ &= \begin{pmatrix} G(x) \\ 0 \end{pmatrix} (y+1) + \begin{pmatrix} G(x + \mu) + G(\mu) + G(0) + L(x) \\ \ell(x) \end{pmatrix} y \\ &= \begin{pmatrix} G(x) \\ 0 \end{pmatrix} + \begin{pmatrix} G(x) + G(x + \mu) + G(\mu) + G(0) + L(x) \\ \ell(x) \end{pmatrix} y \\ &= \begin{pmatrix} G(x) \\ 0 \end{pmatrix} + \begin{pmatrix} (B_\mu + L)(x) \\ \ell(x) \end{pmatrix} y \end{aligned}$$

Since G is at most of algebraic degree 2, we can use Lemma 29, thus B_μ is linear. Therefore $B_\mu + L$ is also linear, so we have T'_μ in the form as in Theorem 38. This means that the tuple $(G, 0, L + B_\mu, \ell)$ yields the APN function T'_μ.

For an element $\nu \in \mathbb{F}_2^n$ we have that M_ν is by definition a linear involution, thus it is an affine bijection, so $M_\nu T \approx_{\mathrm{EA}} T$. We can also rewrite $M_\nu T$, thus we get

$$\begin{aligned}
M_\nu T \begin{pmatrix} x \\ y \end{pmatrix} &= M\nu \left(\begin{pmatrix} G(x) \\ 0 \end{pmatrix} (y+1) + \begin{pmatrix} G(x) + L(x) \\ \ell(x) \end{pmatrix} y \right) \\
&= M_\nu \begin{pmatrix} G(x)(y+1) + G(x)y + L(x)y \\ \ell(x)y \end{pmatrix} \\
&= M_\nu \begin{pmatrix} G(x) + L(x)y \\ \ell(x)y \end{pmatrix} \\
&= \begin{pmatrix} G(x) + L(x)y + \ell(x)y^2\nu \\ \ell(x)y \end{pmatrix} \\
&= \begin{pmatrix} G(x) \\ 0 \end{pmatrix} + \begin{pmatrix} (L+\ell\nu)(x) \\ \ell(x) \end{pmatrix} y
\end{aligned}$$

which is again has the same form as in Theorem 38. Thus, the tuple $(G, 0, L+\ell\nu, \ell)$ yields the APN function $M_\nu T$. We can combine T'_μ and $M_\nu T$ and we get that the tuple $(G, 0, L + B_\mu + \ell\nu, \ell)$ yields an APN function $M_\nu T'_\mu$. Since $T \approx_{\text{EA}} T'_\mu$ and $T \approx_{\text{EA}} M_\nu T$. Therefore $M_\nu T \approx_{\text{EA}} M_\nu T'_\mu$, which leads to $T \approx_{\text{EA}} M_\nu T'_\mu$.

Now we want to prove, that $L + B_\mu + \ell\nu$ are pairwise distinct for $\mu, \nu \in \mathbb{F}_2^n$. This can be proved by showing that the linear mapping

$$\begin{aligned}
J : \mathbb{F}_2^n \times \mathbb{F}_2^n &\to \mathcal{L}(\mathbb{F}_2^n, \mathbb{F}_2^n) \\
\begin{pmatrix} \mu \\ \nu \end{pmatrix} &\mapsto B_\mu + \ell\nu
\end{aligned}$$

is injective since L in $L+B_\mu+\ell\nu$ does not depend on μ, ν. For this, we will show, that the kernel of J is trivial. Suppose, that we have a non-trivial element in the kernel of J. This means that for all $x \in \mathbb{F}_2^n$, we have $B_\mu(x) + \ell(x)\nu = 0$. Now we can only consider all $x \in \mathbb{F}_2^n$ for which $\ell(x) = 1$. This implies $B_\mu(x) = \nu$. We know from the assumptions, that T is an APN function, so by Theorem 38, G is an APN function. Since $B_\mu(x) = G(x) + G(x+\mu) + G(\mu) + G(0) = \nu$ it follows that

$$G(x) + G(x+\mu) = \nu + G(\mu) + G(0)$$

has at most two solutions for every non-zero μ. Thus the only way the equation can be satisfied for all $x \in \mathbb{F}_2^n$ is if μ is a zero vector. Therefore, $B_\mu(x) = G(x) + G(x) + G(0) + G(0)$ is a zero vector. It follows from $B_\mu(x) = \nu$ that ν is a zero vector. So the kernel of J must be trivial.

Now we have to prove the last statement. Let us fix an element $\mu \in \mathbb{F}_2^n$ and let us define $L' := L + B_\mu$. Now we will divide the proof into two cases based on the $\text{rank}(L')$.

In the first case we assume that $\text{rank}(L') = n$. Let us fix a $\nu \in \mathbb{F}_2^n$. We are interested in $\text{rank}(L' + \ell\nu)$. To examine it, we focus on the kernel of $L' + \ell\nu$. Since we are interested in all $x \in \mathbb{F}_2^n$ such that $L'(x) + \ell(x)\nu = 0$, we will divide the proof into two cases based on the value of $\ell(x)$.

In the first case, we focus on all $x \in \mathbb{F}_2^n$ such that $\ell(x) = 0$. Therefore, $L'(x) + \ell(x)\nu = L'(x) = 0$. Since we assume that $\text{rank}(L')$, we know that L' is invertible, so $L'(x) = 0 \iff x = 0$.

In the second case, we focus on all $x \in \mathbb{F}_2^n$ such that $\ell(x) = 1$. Thus, we have

$$\begin{aligned} L'(x) + \nu = 0 &\iff L'(x) = \nu \\ &\iff x = L'^{-1}(\nu) \\ &\iff \ell(L'^{-1}(\nu)) = 1 \end{aligned}$$

Since $\ell \neq 0$, note that $\dim(\mathrm{Ker}(\ell)) = \dim(\mathbb{F}_2^n) - \mathrm{rank}(\ell) = n-1$. Therefore $|\{x \mid \ell(x) = 0\}| = 2^{n-1}$. Since $\mathrm{Im}(\ell) = \{0,1\}$, then $|\{x \mid \ell(x) = 1\}| = 2^n - 2^{n-1} = 2^{n-1}$.

Therefore, if $\nu \in \mathbb{F}_2^n$ is such that $\ell(L'^{-1}(\nu)) = 1$, then $\mathrm{Ker}(L' + \ell\nu) = \{0, L'^{-1}(\nu)\}$. This implies that $\dim(\mathrm{Ker}(L' + \ell\nu)) = 1$. Therefore $\mathrm{rank}(L' + \ell\nu) = \dim(\mathbb{F}_2^n) - \dim(\mathrm{Ker}(L' + \ell\nu)) = n - 1$.

If $\nu \in \mathbb{F}_2^n$ is such that $\ell(L'^{-1}(\nu)) = 0$, then $\mathrm{Ker}(L' + \ell\nu) = \{0\}$. This implies that $\dim(\mathrm{Ker}(L' + \ell\nu)) = 0$. Therefore $\mathrm{rank}(L' + \ell\nu) = \dim(\mathbb{F}_2^n) - \dim(\mathrm{Ker}(L' + \ell\nu)) = n$.

Put it all together, if $\mathrm{rank}(L') = n$, then we have 2^{n-1} possible choices of $\nu \in \mathbb{F}_2^n$ $(\ell(L'^{-1}(\nu)) = 1)$ such that $\mathrm{rank}(L' + \ell\nu) = n-1$ and 2^{n-1} possible choices of $\nu \in \mathbb{F}_2^n$ $(\ell(L'^{-1}(\nu)) = 0)$ such that $\mathrm{rank}(L' + \ell\nu) = n$.

For the second case we assume that $\mathrm{rank}(L') \neq n$. Then the rank is equal to $n-1$ by Lemma 39, since the tuple $(G, 0, L + B_\mu, \ell)$ yields an APN function. Let us choose a $\nu \in \mathbb{F}_2^n$. Again we are interested in $\mathrm{rank}(L' + \ell\nu)$. We want to find its value. To do this we focus on the kernel of $L' + \ell\nu$. Since we are interested in all $x \in \mathbb{F}_2^n$ such that $L'(x) + \ell(x)\nu = 0$, we again divide the proof into two cases based on the value of $\ell(x)$.

In the first case, we focus on all $x \in \mathbb{F}_2^n$ such that $\ell(x) = 0$. Therefore, $L'(x) + \ell(x)\nu = L'(x) = 0$. Let us assume, that x is non-zero vector. Because of the choice of x we know that

$$\begin{pmatrix} L' \\ \ell \end{pmatrix}(x) = \begin{pmatrix} L'(x) \\ \ell(x) \end{pmatrix} = 0.$$

But since x is a non-zero element, this contradicts the results of Lemma 39, because we know from this lemma that the rank of the matrix $\begin{pmatrix} L' \\ \ell \end{pmatrix}$ is n. Therefore, $L'(x) = 0$ is true if and only if $x = 0$.

In the second case, we focus on all $x \in \mathbb{F}_2^n$ such that $\ell(x) = 1$. Thus, we have $L'(x) + \ell(x)\nu = 0 \iff L'(x) = \nu$. Our choice of ν was arbitrary. But since $\nu \in \mathbb{F}_2^n$, we can have either $\nu \notin \mathrm{Im}(L')$ or $\nu \in \mathrm{Im}(L')$. Note that since $n - 1 = \mathrm{rank}(L') = \dim(\mathrm{Im}(L'))$, then $|\mathrm{Im}(L')| = 2^{n-1}$ and therefore $|\{\nu \mid \nu \notin \mathrm{Im}(L')\}| = 2^n - 2^{n-1} = 2^{n-1}$.

If $\nu \notin \mathrm{Im}(L')$, then $L'(x) \neq \nu$, which implies that for all $x \in \mathbb{F}_2^n$ such that $\ell(x) = 1$ we have that $x \notin \mathrm{Ker}(L' + \ell\nu)$. This implies, with the combination of the previous case where $\ell(x) = 0$, that $\mathrm{Ker}(L' + \ell\nu) = \{0\}$, thus $\dim(\mathrm{Ker}(L' + \ell\nu)) = 0$, which implies that $\mathrm{rank}(L' + \ell\nu) = \dim(\mathbb{F}_2^n) - \dim(\mathrm{Ker}(L' + \ell\nu)) = n - 0 = n$.

Now if $\nu \in \mathrm{Im}(L')$, then we know that there exists $y \in \mathbb{F}_2^n$ such that $\nu = L'(y)$. Note that $\dim(\mathrm{Ker}(L')) = \dim(\mathbb{F}_2^n) - \mathrm{rank}(L') = n - (n-1) = 1$. Since $\mathrm{rank}(L') = n - 1$, the L' is not injective. Suppose y is the only solution of $L'(x) = \nu$. This implies that $0 = L'(y) - \nu = L'(y - y)$, thus $\mathrm{Ker}(L') = \{0\}$, but this is a contradiction with $\dim(\mathrm{Ker}(L')) = 1$.

Now let us assume that there exists $x_1 \in \mathbb{F}_2^n$ such that $x_1 \neq y$ and $\nu = L'(y) = L'(x_1)$, then $0 = L'(y - x_1)$. Therefore $\mathrm{Ker}(L') = \{0, y - x_1\}$ and $\mathrm{Ker}(L' + \ell\nu) = \{0, y, x_1, y + x_1\}$ which implies that $\dim(\mathrm{Ker}(L' + \ell\nu)) = 2 \implies \mathrm{rank}(L' + \ell\nu) = n - 2$. This is beyond the scope of this proposition.

Now suppose there exist $x_1, \dots, x_k \in \mathbb{F}_2^n$, for $k \in \mathbb{N}, k \geq 2$ such that they are pairwise distinct and also $y \neq x_i$ for $i \in \{1, \dots, k\}$. This implies that $0, y - x_1, \dots, y - x_k \in \mathrm{Ker}(L')$ which contradicts $\dim(\mathrm{Ker}(L')) = 1$.

Therefore if $\mathrm{rank}(L') \neq n$, then we have 2^{n-1} possible choices of ν ($\nu \notin \mathrm{Im}(L')$) for which $\mathrm{rank}(L' + \ell\nu) = n$. If $\nu \in \mathrm{Im}(L')$, then neither $\mathrm{rank}(L' + \ell\nu) = n$ nor $\mathrm{rank}(L' + \ell\nu) = n - 1$ holds.

□

Using this proposition, we can get a lower bound on the size of the set $\Gamma_{G,\ell}$. Let $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN vectorial Boolean function and let $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$ be a linear mapping such that $\ell \neq 0$. Let us assume, that $\Gamma_{G,\ell} \neq \emptyset$. Then we can choose any $\mu \in \mathbb{F}_2^n$, so we have 2^n possible choices. For those μ we have $2 \cdot 2^{n-1}$ possible choices for the element $\nu \in \mathbb{F}_2^n$. Therefore we have $2^n \cdot (2 \cdot 2^{n-1}) = 2^{2n}$ possible choices for (μ, ν), so $|\Gamma_{G,\ell}| \geq 2^{2n}$.

We can give the following definition and proposition, which gives us a more elegant way of denoting linear functions $L + B_\mu + \ell\nu$ for which we have EA-equivalent functions to T from Proposition 40. The following definition is based on the text from [5, Section 5]. Proposition 41 follows from the text at the end of the first section of [5, Section 5].

Definition 44. *Let $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN vectorial Boolean function and let $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$ be linear mapping such that $\ell \neq 0$. Let $L, L' \in \Gamma_{G,\ell}$. We say that L and L' are* Γ-equivalent *if there exist $\mu, \nu \in \mathbb{F}_2^n$ such that $L' = L + B_\mu + \ell\nu$.*

Proposition 41. *Let $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN vectorial Boolean function and let $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$ be linear mapping such that $\ell \neq 0$. Let $L, L' \in \Gamma_{G,\ell}$ be Γ-equivalent. Then the tuples $(G, 0, L, \ell)$ and $(G, 0, L', \ell)$ yield EA-equivalent APN functions.*

Proof. This proposition was proved in Proposition 40.

□

5.4 Algorithm

The algorithm for finding quadratic APN functions with maximum linearity is presented in [5, Section 5.1] as Algorithm 1. We will describe the algorithm and its functions and put them in the context of the theory presented in this chapter. The idea behind the algorithm is that it takes as input a quadratic APN vectorial Boolean function $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and for all $\ell : \mathbb{F}_2^n \to \mathbb{F}_2$ it tries to find at least one suitable L from $\Gamma_{G,\ell}$. If it finds such a L, it calculates the APN function $T : \mathbb{F}_2^{n+1} \to \mathbb{F}_2^{n+1}$ from Theorem 38 and returns it as output.

All of the algorithms are implemented in the Python 3 programming language using Sagemath, but in this text we will use pseudo-code for better understanding. The implementation of the algorithm is attached in A.13.

The core of the algorithm follows the algorithm from [5, Attachment B], thus the functions `Gen_System`, `Is_Extendable` and `Extensions` and Algorithm 5 are taken from [5]. We will use these functions and describe them in more detail and in the context of the theory presented.

5.4.1 Description

Now we will examine functions from that algorithm. Let us start with the function `Gen_System`. The goal of this function is to find L that satisfies condition 2 from Theorem 38. Let us recall this condition:

$$\langle \pi_G(\alpha), L(\alpha)\rangle = 1 \text{ for all } \alpha \in \mathbb{F}_2^n \setminus \{0\} \text{ with } \ell(\alpha) = 0.$$

The function takes as input a quadtratic APN function $G : \mathbb{F}_2^n \to \mathbb{F}_2^n$, the ortho-derivative π_G and a function ℓ (which is a vector from $\mathbb{F}_2^n$). The output is a matrix M with k rows and n^2 columns, where k is the number of $\alpha \in \mathbb{F}_2^n$ that satisfy $\langle \ell, \alpha\rangle = 0$. If this matrix is solvable, then the solution will give us a mapping L in the function `Is_Extendable`. Then we will present the implementation of the algorithm following the pseudo-codes of the functions

Because of the input, we have fixed π_G and ℓ. Condition 2 from Theorem 38 is only true for non-zero α, which satisfies $\ell(\alpha) = 0$, which is equivalent to $\langle \ell, \alpha\rangle = 0$. Let us assume for the moment that only one α satisfies this. We want to find L such that $\langle \pi_G(\alpha), L(\alpha)\rangle = 1$. Let us denote L and α and their elements by

$$L = \begin{pmatrix} l_{11} & l_{12} & \cdots & l_{1n} \\ l_{21} & l_{22} & \cdots & l_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{pmatrix} \qquad \alpha = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix}$$

and $\pi_G(\alpha)$ and its elements as

$$(\pi_G(\alpha))^T = (\pi_G^1(\alpha), \pi_G^2(\alpha), \dots, \pi_G^n(\alpha))^T.$$

Thus, we can write

$$L(\alpha) = \begin{pmatrix} l_{11} & l_{12} & \cdots & l_{1n} \\ l_{21} & l_{22} & \cdots & l_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{pmatrix} \cdot \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} l_{11}\alpha_1 + l_{12}\alpha_2 + \cdots + l_{1n}\alpha_n \\ l_{21}\alpha_1 + l_{22}\alpha_2 + \cdots + l_{2n}\alpha_n \\ \vdots \\ l_{n1}\alpha_1 + l_{n2}\alpha_1 + \cdots + l_{nn}\alpha_n \end{pmatrix}$$

and therefore

$$\langle \pi_G(\alpha), L(\alpha)\rangle = (\pi_G^1(\alpha), \pi_G^2(\alpha), \dots, \pi_G^n(\alpha)) \cdot \begin{pmatrix} l_{11}\alpha_1 + l_{12}\alpha_2 + \cdots + l_{1n}\alpha_n \\ l_{21}\alpha_1 + l_{22}\alpha_2 + \cdots + l_{2n}\alpha_n \\ \vdots \\ l_{n1}\alpha_1 + l_{n2}\alpha_1 + \cdots + l_{nn}\alpha_n \end{pmatrix}$$

Using the property of matrix multiplication, we can rewrite the last equation as follows. Since the notation is long, we will define M_α as

$$M_\alpha := \left(\pi_G^1(\alpha)\alpha_1, \dots, \pi_G^1(\alpha)\alpha_n, \pi_G^2(\alpha)\alpha_1, \dots, \pi_G^2(\alpha)\alpha_n, \dots, \dots, \pi_G^n(\alpha)\alpha_n\right)$$

Therefore we have

$$\langle \pi_G(\alpha), L(\alpha) \rangle = M_\alpha \cdot (l_{11}, l_{12}, \dots, l_{1n}, l_{21}, \dots, l_{2n}, \dots, l_{n1}, \dots, l_{nn})^T$$

Let us denote the vector with the coefficients l_{ij} as $\widetilde{L}$, thus we have

$$1 = \langle \pi_G(\alpha), L(\alpha) \rangle = M_\alpha \cdot \widetilde{L}.$$

This means that if we want to find $\widetilde{L}$ (which means that we want to find L), we have to solve the equation $M_\alpha(\widetilde{L}) = 1$.

We have assumed, that we have only one α which satisfies $\langle \ell, \alpha \rangle = 0$. Now consider all such α. Suppose the number of such α is k. We can then do all of this for each of those α, therefore we can define a matrix M, which will have k rows consisting of M_α for those α. Therefore we have the equation $M \cdot \widetilde{L} = 1$, where $1 \in \mathbb{F}_2^k$. Therefore, if we solve the equation $M \cdot \widetilde{L} = 1$, we get the matrix L which we want. So let us put all of this into the context of the `Gen_System` function. As mentioned, we are only interested in α satisfying $\langle \ell, \alpha \rangle = 0$. This is the reason for the condition on the line 4. Then on the lines 6-8 we generate M_α and on the line 11 we put this M_α into the matrix M.

```
 1: function Gen_System("G","$\pi_G$","$\ell$"):
 2:     "M" ← initialised as an empty list
 3:     for $\alpha \in [1, 2^n - 1]$ do
 4:         if $\langle \ell, \alpha \rangle = 0$ then
 5:             "M$_\alpha$" ← initialised as an empty list
 6:             for $bit\_\pi \in [0, \dots, n-1]$ do
 7:                 for $bit\_\alpha \in [0, \dots, n-1]$ do
 8:                     "M$_\alpha$" ← append (bit on $bit\_\alpha$ position in $\alpha$)·(bit on $bit\_\pi$ position in "$\pi_G$"[$\alpha$]) to the list
 9:                 end for
10:             end for
11:             "M" ← append "M$_\alpha$" to the list
12:         end if
13:     end for
14:     return(matrix $M$)
15: end function
```

As we mentioned before, the function `Is_Extendable` will give us the linear mapping L from Theorem 38 based on the output of the function `Gen_System`.Now we will examine the function `Is_Extendable`.

On the line 2, we generate the matrix M (from the function `Gen_System`). This matrix, if $M \cdot \widetilde{L} = 1$ from the description of the function `Gen_System` is solvable, will give us the solution $\widetilde{L}$, which contains elements from the matrix of the linear mapping L. Therefore, if the matrix M is solvable, then we have our desired linear mapping L. Thus we check this on the line 4.

The equation $M \cdot \widetilde{L} = 1$ can have more solutions. We want to decide whether they are Γ-equivalent or not. We will do this with some uncertainty. From Proposition 40, we know that if $\Gamma_{G,\ell} \neq \emptyset$, then $|\Gamma_{G,\ell}| \geq 2^{2n}$. From the definition of Γ-equivalence we know, that $L, L' \in \Gamma_{G,\ell}$ are Γ-equivalent if there exist $\mu, \nu \in \mathbb{F}_2^n$ such that $L' = L + B_\mu + \ell\nu$. Thus the number of Γ-equivalent linear mappings to L is 2^{2n}. Therefore, if we have more than 2^{2n} solutions of $M\widetilde{L} = 1$, then we have $|\Gamma_{G,\ell}| > 2^{2n}$ and $|\operatorname{Im}(M)| < 2^{n^2} - 2^{2n}$. Thus

$$|\operatorname{Im}(M)| < 2^{n^2} - 2^{2n} = 2^{2n}\left(2^{n^2-2n} - 1\right) < 2^{2n}\left(2^{n^2-2n}\right) < 2^{n^2-2n}.$$

This implies that $\dim(\operatorname{Im}(M)) < n^2 - 2n$. Hence

$$\dim(\operatorname{Ker}(M)) = \dim(F_2^{n^2}) - \dim(\operatorname{Im}(M)) > n^2 - (n^2 - 2n) = 2n.$$

Using this inequality, we can check on the line 9, if there might be some Γ-inequivalent solutions by calculating the dimension of the kernel of M.

Then the function transforms the $\widetilde{L}$ into the 2-dimensional array and computes the function T from the Theorem 38. The function T is from $\mathbb{F}_2^{n+1}$to $\mathbb{F}_2^{n+1}$. As in the Theorem 38, let the input of T be (x, y), where $x \in \mathbb{F}_2^n$ and $y \in \mathbb{F}_2$. The first for loop on the line 14 adds values for all $(x, 0)$ to the function (which we understand as a look-up table). The second for loop on the line 17 adds values to the function for all $(x, 1)$. Therefore, at the end of this function, the array T contains 2^{n+1} elements.

```
 1: function Is_Extendable("G","$\pi_G$","$\ell$"):
 2:     "M" ← Gen_System("G","$\pi_G$","$\ell$")
 3:     "v$_1$" ← vector with ones on $k$ positions, where $k$ is the number of rows of
    "M"
 4:     if $M\widetilde{L} = v_1$ has at least one solution then
 5:         solve: $M\widetilde{L} = v_1$
 6:     else
 7:         return(empty list)
 8:     end if
 9:     if dim(Ker($M$))$> 2n$ then
10:         Inform the user that there may be other $\Gamma$-inequivalent solutions.
11:     end if
12:     "L" ← transform $\widetilde{L}$ into the 2-dimensional array
13:     "T" ← initialised as an empty list
14:     for $i \in [0, \dots, 2^n - 1]$ do
15:         "T" ← append list with integer representation of: $G[i]$ shifted by one
    bit to the left
16:     end for
17:     for $i \in [0, \dots, 2^n - 1]$ do
18:         "T" ← append list with integer representation of: $(G[i] \oplus L[i]$ shifted
    by one bit to the left) $\oplus \langle \ell, i \rangle$
19:     end for
20:     return("T")
21: end function
```

The last function is `Extensions`. It's purpose is to find all functions T from Theorem 38 for a given function G. The function gets a function G as an input. First it calculates the ortho-derivative π_G. Then, for all possible choices of ℓ, it tries to find the extension of G in standard form using `Is_Extendable` on the line 5. If it succeeds, the function adds the extension to the list "sol".

```
 1: function Extensions("G"):
 2:     "sol" ← initialised as an empty list
 3:     "$\pi_G$" ← compute ortho-derivative of $G$
```

```
4:     for $\ell \in [1, \ldots, 2^n - 1]$ do
5:         "T" $\leftarrow$ Is_Extendable("G","$\pi_G$","$\ell$")
6:         if "T" is not an empty list then
7:             "sol" $\leftarrow$ append "T" to the list
8:         end if
9:     end for
10:    return("sol")
11: end function
```

Now we can finally examine the Algorithm 1 from [5]. Since we described the function `Extensions`, the description of the algorithm itself is very straightforward. It take function from the input set $\mathcal{G}$ and use it as an input to the function `Extensions`. If the function `Extensions` found $n + 1$-bit function T, it add it to the list "solutions". After examination of all $G \in \mathcal{G}$, the algorithm shows founded functions.

Algorithm 5 Classification of $n+1$-bit quadratic APN functions with maximum linearity up to EA-equivalence

Input: set $\mathcal{G}$ which contains n-bit quadratic APN vectorial Boolean functions
Output: $n + 1$-bit quadratic APN functions with maximum linearity

```
1: "solutions" $\leftarrow$ initialised as an empty array
2: for $i \in [0, \ldots, |\mathcal{G}|]$ do
3:     "T" $\leftarrow$ Extensions($\mathcal{G}$)
4:     if "T" is not empty then
5:         "solutions" $\leftarrow$ append "T" to the list
6:     end if
7: end for
8: Show all functions from the array "solutions".
```

5.4.2 Results

The results are attached in A.14.

n=7

First we try to replicate the results of [5]. Let us denote $\mathcal{G} = \{G_1, \ldots, G_{488}\}$ which is the set of all 7-bit quadratic APN functions found in the [13]. All of these functions are pairwise EA-inequivalent as it is stated in [5]. The values of these functions can be found in [14] in a file `sevenBitAPN.py`. For each of these functions we run Algorithm 5.

The output of the algorithm is four quadratic APN functions with maximum linearity in the form of a look-up table. These four functions are pairwise EA-inequivalent. This is because all of the functions in the file `sevenBitAPN.py` are EA-inequivalent.

Application to results from Chapter 2.

We have used the results of Subsection 2.3.3 (A.8). By representative of an EA-equivalence class, we mean the first function in the class, as given in A.8.

For $n = 2$ we know that there is only one EA-equivalence class. We applied the algorithm to the representative of this class and got a function as an output. For $n = 3$ we have only one EA-equivalence class. We applied the algorithm to the representative of this class and got a function as an output. For $n = 4$ we have only one EA-equivalence class. After applying the algorithm to the representative, we did not get a function as an output. For $n = 5$ we know that there are two EA-equivalence classes. After applying the algorithm to these representatives we obtain a function for each of them. For $n = 6$ we have 13 EA-equivalence classes. We applied the algorithm to representative of each of these classes, but we do not get a function for any of these representatives.

Conclusion

This thesis intended to provide a better understanding of the methods for finding APN vectorial boolean functions and to classify them into equivalence classes which were presented in several recent articles. We mathematically expanded the theory from these articles, extended and clarified proofs, and provided the implementation of the algorithm with explanations in the context of the presented theory. We were also able to reproduce the results of the papers, demonstrating the effectiveness of the methods.

The methods presented in this thesis can be used in higher dimensions if useful constraints are found, providing a direction for future research.

Bibliography

[1] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, page 212–219, New York, NY, USA, 1996. Association for Computing Machinery. ISBN 0897917855. doi: 10.1145/237814.237866.

[2] Christof Beierle and Gregor Leander. New Instances of Quadratic APN Functions. *IEEE Transactions on Information Theory*, 68(1):670–678, 2022. doi: 10.1109/TIT.2021.3120698.

[3] Christof Beierle, Marcus Brinkmann, and Gregor Leander. Linearly Self-Equivalent APN Permutations in Small Dimension. *IEEE Transactions on Information Theory*, 67(7):4863–4875, 2021. doi: 10.1109/TIT.2021.3071533.

[4] David S. Dummit and Richard M. Foote. *Abstract algebra.* Third edition. John Wiley and Sons, Inc., 2004. ISBN 0-471-43334-9.

[5] Christof Beierle, Gregor Leander, and Léo Perrin. Trims and extensions of quadratic APN functions. *Designs, Codes and Cryptography*, 90:1009–1036, 03 2022. doi: 10.1007/s10623-022-01024-4.

[6] Faruk Göloglu. Lecture notes on Boolean functions. Faculty of Mathematics and Physics Charles University, 2020.

[7] David Stanovský. *Základy algebry.* MatfyzPress, 2010. ISBN 978-80-7378-105-7.

[8] Claude Carlet. *Boolean Functions for Cryptography and Coding Theory.* Cambridge University Press, 2020. ISBN 978-1-108-47380-4. doi: 10.1017/9781108606806.

[9] Satoshi Yoshiara. Equivalences of power APN functions with power or quadratic APN functions. *Journal of Algebraic Combinatorics*, 44:561–585, 3 2016. doi: 10.1007/s10801-016-0680-z.

[10] Morgan Barbier, Hayat Cheballah, and Jean-Marie Le Bars. On the computation of the Möbius transform. *Theoretical Computer Science*, 809:171–188, 2020. ISSN 0304-3975. doi: https://doi.org/10.1016/j.tcs.2019.12.005.

[11] Marcus Brinkmann and Gregor Leander. On the classification of APN functions up to dimension five. *Designs, Codes and Cryptography*, 49:273–288, 3 2008. doi: 10.1007/s10623-008-9194-6.

[12] Anne Canteaut, Alain Couvreur, and Léo Perrin. Recovering or Testing Extended-Affine Equivalence. *IEEE Transactions on Information Theory*, 68(9):6187–6206, 2022. doi: 10.1109/TIT.2022.3166692.

[13] Konstantin Kalgin and Valeriya Idrisova. The classification of quadratic APN functions in 7 variables and combinatorial approaches to search for APN functions. 15:239–256, 2023. doi: 10.1007/s12095-022-00588-1.

[14] Jules Baudrin, Aurélien Boeuf, Alain Couvreur, Mathias Joly, and Léo Perrin. SboxU: Tools for studying S-boxes v1.3, 2023. URL `https://github.com/lpp-crypto/sboxU`.

[15] Marco Calderini. On the EA-classes of known APN functions in small dimensions. *Cryptography and Communications*, 12:821–840, 4 2020. doi: 10.1007/s12095-020-00427-1.

[16] Anne Canteaut and Léo Perrin. On CCZ-equivalence, extended-affine equivalence, and function twisting. *Finite Fields and Their Applications*, 56:209–246, 2019. ISSN 1071-5797. doi: 10.1016/j.ffa.2018.11.008.

[17] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-correcting Codes*, volume 16. North-Holland Publishing Company, 1977. ISBN 0-444-85010-4.

A. Attachments

To run the algorithms from A.1 to A.6, Python 3, Sagemath and the Cython Module for Python and also the sboxU library [14] must be installed. The ".py" files are also attached to the thesis in the form of ".txt" files. All functions in the source codes includes comments. The following Python source codes and results are available at `https://github.com/arskn/master_thesis/` in the mentioned folders.

A.1 Attachment one - Python 3 Implementation of the Algorithm from Subsection 2.3.1

The Python source code is available as `chapter2-algorithm1-A1.py` in the folder `chapter2`.

A.2 Attachment two - Python 3 Implementation of the Algorithm from Subsection 2.3.1 - modified

The Python source code is available as `chapter2-algorithm1-A2.py` in the folder `chapter2`.

A.3 Attachment three - Python 3 Implementation of the Algorithm from Subsection 2.3.2

The Python source code is available as `chapter2-algorithm1-A3.py` in the folder `chapter2`.

A.4 Attachment four - Python 3 Implementation of the Algorithm from Subsection 2.3.2 - modified

The Python source code is available as `chapter2-algorithm1-A4.py` in the folder `chapter2`.

A.5 Attachment five - Results from Subsection 2.3.2

The results are available in the folder `chapter2/results-A5`.

- for $n = 2$ as `chapter2-algorithm1-A3-output-n2.py`

- for $n = 3$ as `chapter2-algorithm1-A3-output-n3.py`
- for $n = 4$ as `chapter2-algorithm1-A4-output-n4.py`
- for $n = 5$ as `chapter2-algorithm1-A4-output-n5.py`
- for $n = 6$ as `chapter2-algorithm1-A4-output-n6.py`
- for $n = 7$ as `chapter2-algorithm1-A4-output-n7.py`

A.6 Attachment six - Python 3 Implementation of the Algorithm from Subsection 2.3.3

The Python source code is available as `chapter2-algorithm2-A6.py` in the folder `chapter2`.

A.7 Attachment seven - Python 3 Implementation of the Algorithm from Subsection 2.3.3 - modified

Running the algorithm requires Python 3, Sagemath, the Cython Module for Python, the sboxU library [14] and the Python code `Equivalence.py` from [12]. The Python source code is available as `chapter2-algorithm2-A7.py` in the folder `chapter2`.

A.8 Attachment eight - Results from Subsection 2.3.3

The results are available in the folder `chapter2/results-A8`

- for $n = 5$ as `chapter2-algorithm2-A7-output-n5.txt`
- for $n = 6$ as `chapter2-algorithm2-A6-output-n6.txt`
- for $n = 7$ as `chapter2-algorithm2-A6-output-n7.txt`

A.9 Attachment nine - Python 3 Implementation of Algorithm from Chapter 3

To run the algorithm Python 3, Sagemath and the Cython Module for Python must be installed. The Python source code is available in the folder `chapter3` as `chapter3 -algorithm3-A9.py`.

A.10 Attachement ten - Results from Chapter 3

The results are available in the folder `chapter3/results-A10`

- for $n = 2$ as `chapter3-algorithm3-A9-output-n2.txt`
- for $n = 3$ as `chapter3-algorithm3-A9-output-n3.txt`
- for $n = 4$ as `chapter3-algorithm3-A9-output-n4.txt`
- for $n = 5$ as `chapter3-algorithm3-A9-output-n5.txt`
- for $n = 6$ as `chapter3-algorithm3-A9-output-n6.txt`
- for $n = 7$ as `chapter3-algorithm3-A9-output-n7.txt`
- for $n = 8$ as `chapter3-algorithm3-A9-output-n8.txt`
- for $n = 9$ as `chapter3-algorithm3-A9-output-n9.txt`
- for $n = 10$ as `chapter3-algorithm3-A9-output-n10.txt`
- for $n = 11$ as `chapter3-algorithm3-A9-output-n11.txt`
- for $n = 12$ as `chapter3-algorithm3-A9-output-n12.txt`

A.11 Attachment eleven - Python 3 Implementation of the Algorithm from Chapter 4

To run the algorithm Python 3 and the sboxU library [14] must be installed. The Python source code is available as `chapter4-algorithm4-A11.py` in the folder `chapter4`.

A.12 Attachment twelve - Results from Chapter 4

The results are available in the folder `chapter4/results-A12`

- for $n = 7$ to 6 as `chapter4-algorithm4-A11-output-n7.txt`
- for $n = 6$ to 5 as `chapter4-algorithm4-A11-output-n6.txt`
- for $n = 5$ to 4 as `chapter4-algorithm4-A11-output-n5.txt`

A.13 Attachment thirteen - Python 3 Implementation of the Algorithm from Chapter 5 in Python 3

To run the algorithm Python 3, Sagemath, the Cython Module for Python and the sboxU library [14] must be installed. The Python source code is available as `chapter5-algorithm5-A13.py` in the folder `chapter5`.

A.14 Attachment fourteen - Results from Chapter 5

The results are available in the folder `chapter5/results-A14` as `chapter5-algorithm5-A13-output.txt`.