Recovering or Testing Extended-Affine Equivalence

Anne Canteaut, Alain Couvreur, Léo Perrin

February 26, 2021

Abstract

Extended Affine (EA) equivalence is the equivalence relation between two vectorial Boolean functions F and G such that there exist two affine permutations A, B, and an affine function C satisfying $G = A \circ F \circ B + C$. While *a priori* simple, it is very difficult in practice to test whether two functions are EA-equivalent. This problem has two variants: *EA-testing* deals with figuring out whether the two functions can be EA-equivalent, and *EA-recovery* is about recovering the tuple (A, B, C) if it exists.

In this paper, we present a new efficient algorithm that efficiently solves the EA-recovery problem for quadratic functions. Though its worst-case complexity is obtained when dealing with APN functions, it supersedes all previously known algorithms in terms of performance, even in this case. This approach is based on the Jacobian matrix of the functions, a tool whose study in this context can be of independent interest.

In order to tackle EA-testing efficiently, the best approach in practice relies on class invariants. We provide an overview of the literature on said invariants along with a new one based on the *ortho-derivative* which is applicable to quadratic APN functions, a specific type of functions that is of great interest, and of which tens of thousands need to be sorted into distinct EA-classes. Our ortho-derivative-based invariant is both very fast to compute, and highly discriminating.

1 Introduction

Affine equivalence, and its generalization named *extended affine equivalence*, are two equivalence relations between vectorial Boolean functions defined as follows.

Definition 1 ((Extended) Affine Equivalence)**.** *Two vectorial Boolean functions $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and $G : \mathbb{F}_2^n \to \mathbb{F}_2^m$ are* affine equivalent *if $G = A \circ F \circ B$ for some affine permutations A of $\mathbb{F}_2^n$ and B of $\mathbb{F}_2^m$. They are* extended affine equivalent *(EA-equivalent) if $G = A \circ F \circ B + C$ where A and B are as before and where $C : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is an affine function.*

Determining whether two functions $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^m$ are EA-equivalent is a problem which appears in several situations, for instance when classifying vectorial Boolean functions (since most relevant cryptographic parameters of the functions are invariant under EA-equivalence) or in cryptanalysis. In this paper, we will focus on the following problem, named *EA-recovery*, and on its variant.

Problem 1 (EA-recovery)**.** *Let F and G be two functions from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$. Find, if they exist, three affine mappings A, B, C such that $G = A \circ F \circ B + C$.*

In practice, the following variant of Problem 1 better captures some situations that occur in particular when classifying vectorial Boolean functions with good cryptographic properties.

Problem 2 (EA-testing)**.** *Let $\{F_i\}_{0 \leqslant i < \ell}$ be a large set of ℓ functions from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$. Partition this set in such a way that two functions in distinct subsets are not EA-equivalent.*

For example, if a large number ℓ of quadratic APN functions are generated as was done in [YWL14] and [BL20a], then we may want to check whether these can be clustered into EA-equivalence classes. Of course, the second problem can be solved using a solution to the first one as a subroutine. Thus, in order for a solution to be interesting, it should not be quadratic in ℓ.

1.1 State of the Art: Affine Equivalence Recovery in some Specific Cases

When $C \neq 0$, very few results are known. In [BK12], Budaghyan and Kazymyrov present algorithms for solving several particular cases, when A, B or C have a specific form.

The case $C = 0$ (corresponding to *affine equivalence*) has been solved when F and G are permutations in the sense that we have algorithms capable of finding A and B in this context. However, we can hope both to improve the efficiency of these algorithms and to solve the case where F and G are not permutations. Let us nevertheless recall the known results on *affine equivalence*.

1.1.1 Guess-and-Determine

The first such algorithm was proposed in [BDBP03]. It is based on a subroutine which returns the "linear representative" of a permutation. Given a permutation F, it returns the two linear permutations L_0 and L_1 such that $L_1 \circ F \circ L_0$ is the smallest in the lexicographic order. This algorithm is based on a guess-and-determine approach. Its authors estimated its time complexity to be $\mathrm{O}\left(n^3 2^n\right)$ if $F(0) \neq 0$ and $\mathrm{O}\left(n^3 2^{2n}\right)$ otherwise.

We have implemented this algorithm and, in practice, it can be worse than this. Indeed, the complexity analysis assumes that a contradiction in the guess-and-determine will occur fast enough. It is usually true but, in some cases, it may happen that we end up having to loop through all values for another variable. Performance is thus hit with another factor 2^n.

Using this algorithm, it is easy to recover A and B when they are linear. However, when they are affine, we also need to brute-force the constants. In this case, we generate two lists containing the linear representatives of $x \mapsto F(x \oplus a)$ and $x \mapsto b \oplus G(x)$. We then look for a match in these lists. The cost in this case is multiplied by 2^n in both time and memory. The overall time is then $\mathrm{O}\left(n^3 2^{2n}\right)$ (assuming that the complexity estimation of the authors of [BDBP03] is correct).

The advantage of this method is that it works for all permutations. The downsides are that its complexity can be underestimated and that it only works for permutations.

1.1.2 Rank Table

In a more recent paper [Din18], Dinur proposed a completely different approach based on so-called "rank tables". Paraphrasing the introduction of said paper, the main idea of the algorithm is to compute the rank tables of both F and G and then use these tables to recover the affine transformation B, assuming that $G = A \circ F \circ B$. The rank tables of F and G are obtained as follows. We derive from F (resp. from G) several functions, each one defined by restricting its 2^n inputs to an affine subspace of dimension $n-1$. Since each such derived function has an associated rank, we assign to each possible $(n-1)$-dimensional subspace a corresponding rank. As there are $(2^{n+1}-2)$ possible affine subspaces, we obtain $(2^{n+1}-2)$ rank values for F (resp. for G). These values are collected in the rank table of F (resp. G), where a rank table entry r stores the set of all affine subspaces assigned to rank r. We then look for matches in these two rank tables.

This approach is faster as the computational time for solving the affine-equivalence case is $\mathrm{O}\left(n^3 2^n\right)$, i.e. it is 2^n times faster than the algorithm of Biryukov et al. Unlike the latter, the rank table-based approach works even if the functions are not bijective; but it does require that their algebraic degree is high enough, i.e. $n-1$ or $n-2$ [Din18]. We have used an implementation of this algorithm by its author and we have confirmed that it could very efficiently handle non-bijective functions of degree $n-1$. However, for functions of degree $n-2$, we have found it to fail.

1.2 Our Results

While the two previously mentioned algorithms are dedicated to *affine-equivalence* recovery, i.e, to the case $C=0$, we present here the first efficient algorithm for *EA-equivalence* recovery when the involved functions F and $G : \mathbb{F}_2^n \to \mathbb{F}_2^m$ are quadratic. We prove that its complexity depends on the differential spectrum of the function and is estimated to be of $\mathrm{O}\left(R^s(m^2+n^2)^\omega\right)$, where ω denotes the complexity exponent of operations in linear algebra and R is the number of vectors $v \in \mathbb{F}_2^n \setminus \{0\}$ at which the rank of the Jacobian matrix is the smallest possible. The last parameter s is a number of guesses which, when $m=n$, can be chosen to be equal to 3. Hence, the estimated complexity is $\mathrm{O}\left(R^3 n^{2\omega}\right)$ and it turns out that for random Boolean functions the quantity R is frequently very small. On the other hand, the most difficult case corresponds to the case of APN functions where the complexity is of $\mathrm{O}\left(2^{2n}(m^2+n^2)^\omega\right)$.

The second part of the paper details several tools for solving the *EA-testing problem* for functions of any degree. Most notably, we propose some new and very efficient EA-invariants for quadratic APN functions, which is a major use-case for this problem. These techniques are then used to partition the CCZ-classes of all the 6-bit quadratic APN functions into EA-classes. Also, by applying it to 8-bit quadratic APN functions, we show that this method is by far the most efficient one for solving Problem 2 in the case of quadratic APN functions.

It is worth noticing that, as detailed in Table 1, only some problems related to EA-equivalence have been solved. In fact, finding a general and efficient algorithm for EA-recovery for functions of degree strictly greater than two remains an open problem. Several simplified cases have been solved in [BK12], where typically some affine functions are only constant additions. Some algorithms from that paper involve more complex restrictions, and are not listed below.

Condition	Complexity	Reference
$C=0$, $m=n$, Permutation	$\mathrm{O}\left(n^3 2^{2n}\right)$	[BDBP03]
$C=0$, $\deg(F) \geqslant n-1$	$\mathrm{O}\left(n^3 2^n\right)$	[Din18]
$A(x)=x\oplus a, B(x)=x\oplus b$	$\mathrm{O}\left(n2^n\right)$	[BK12]
$B(x)=x\oplus b$	$\mathrm{O}\left(m2^{3n}\right)$	[BK12]
$\deg(F)=2$	$\mathrm{O}\left(n^{2\omega}2^{2n}\right)$	Section 3.4

Table 1: Algorithms solving the EA-recovery of $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and $A \circ F \circ B + C$. See Section 4.1 for an overview of EA-testing.

Organization of the Paper. We first recall the basic concepts and definitions needed in Section 2.

The rest of the paper successively presents our algorithms that can efficiently tackle both Problems 1 and 2. First, we show how to reduce EA-recovery to the resolution of a linear system using the Jacobian matrix when the functions under consideration are quadratic. This approach is described in Section 3. While it is only applicable to quadratic functions, the corresponding algorithm is efficient and recovers the full triple (A,B,C).

Then, we describe a general approach based on class invariants which can solve the EA-testing (Problem 2). We list all the CCZ- and EA-class invariants we are aware of from the literature and, in the case of quadratic APN functions, we introduce a new one based on *ortho-derivatives*. While it is only defined for quadratic APN functions, this case is of great practical importance: for instance, constructing quadratic APN functions is of interest for finding APN permutations operating on an even number of variables, as the only known example of such a permutation is derived from a quadratic APN function by CCZ-equivalence [BDMW10]. More importantly, the corresponding CCZ-class invariants are very fine grained, and can efficiently prove that more than $20,000$ distinct quadratic APN functions of 8 variables fall into different CCZ-class in only a few minutes on a regular desktop computer.

Our optimized implementations of all these invariants are available within the Sage package `sboxU`.[1]

2 Preliminaries and Definitions

We consider vectorial Boolean functions, that is functions mapping $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ for some non-zero m and n. Any such function is equivalently seen as a collection of m Boolean functions $\mathbb{F}_2^n \to \mathbb{F}_2$, called its *coordinates*. The following notions will be extensively used thorough the paper.

Differential Properties. The resilience of a function to *differential attacks* depends on properties of its derivatives.

Definition 2 (Derivative)**.** *Let F be a function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$. The* derivative of F with respect to $a \in \mathbb{F}_2^n$ *is the function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$ defined by*

$$\Delta_a F : x \in \mathbb{F}_2^n \mapsto F(x+a) + F(x).$$

In practice, these properties are analyzed through the following values, corresponding to the entries of its *difference distribution table (DDT)*.

Definition 3. *Let F be a function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$. The DDT of F is the $2^n \times 2^m$ array consisting of all elements*

$$\delta_F(a,b) = \#\{x \in \mathbb{F}_2^n \ : F(x+a) + F(x) = b\}, \forall (a,b) \in \mathbb{F}_2^n \times \mathbb{F}_2^m \ .$$

The differential uniformity *of F [Nyb94] is defined as*

$$\delta(F) = \max_{a \neq 0, b} \delta_F(a,b) \ ,$$

and the differential spectrum *is the multi-set*

$$\{\delta_F(a,b), a \in \mathbb{F}_2^n, b \in \mathbb{F}_2^m\} \ .$$

Obviously, $\delta(F) \geqslant 2^{n-m}$ and the functions for which equality holds are named *Perfect Nonlinear* or *bent*. Such functions exist only when n is even and $m \leqslant n/2$ [Nyb91]. When $m \geqslant n$, it satisfies $\delta(F) \geqslant 2$ and the functions for which equality holds are named *Almost Perfect Nonlinear (APN)* functions.

[1] `sboxU` is available for download at https://github.com/lpp-crypto/sboxU.

Walsh Transform. Similarly, the resistance of a function to linear attacks is evaluated through its *Linear approximation table (LAT)*, whose entries are given by the Walsh transform.

Definition 4 (Walsh transform)**.** *Let F be a function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$. Its Walsh transform at $(a,b) \in \mathbb{F}_2^n \times \mathbb{F}_2^m$ is the signed integer defined by*

$$\mathcal{W}_F(a,b) = \sum_{x \in \mathbb{F}_2^n} (-1)^{a \cdot x + b \cdot F(x)}$$

where $y \cdot z$ denotes the canonical inner product on $\mathbb{F}_2^n$, i.e. *$y \cdot z := \sum_{i=1}^{n} y_i z_i$. The* Walsh spectrum *of F is then the multi-set*

$$\{\mathcal{W}_F(a,b), a \in \mathbb{F}_2^n, b \in \mathbb{F}_2^m\}\,.$$

Degree. The degree of a vectorial function is then defined as follows.

Definition 5 (Degree)**.** *Let F be a function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$. The* degree *of F is the maximal degree of the algebraic normal forms of its coordinates.*

Quadratic functions, *i.e.* functions of degree 2, will play an important role in the paper.

CCZ-Equivalence. While this paper focuses on EA-equivalence, there exists a more general notion of equivalence between vectorial Boolean functions defined by Carlet, Charpin and Zinoviev [CCZ98] and called *CCZ-equivalence*. This notion will be widely used in Section 4.

Definition 6. *Two functions $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and $G : \mathbb{F}_2^n \to \mathbb{F}_2^m$ are CCZ-equivalent if there exists an affine permutation $\mathcal{A}$ of $\mathbb{F}_2^n \times \mathbb{F}_2^m$ such that*

$$\mathcal{A}\left(\{(x, F(x)), x \in \mathbb{F}_2^n\}\right) = \{(x, G(x)), x \in \mathbb{F}_2^n\}\,.$$

Obviously, two functions which are EA-equivalent are also CCZ-equivalent, but the converse does not hold.

In general, given a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ and an affine permutation $\mathcal{A}$ of $\mathbb{F}_2^n \times \mathbb{F}_2^m$, there is *a priori* no function G such that

$$\mathcal{A}\big(\,\{(x, F(x)), \forall x \in \mathbb{F}_2^n\}\,\big) = \{(x, G(x)), \forall x \in \mathbb{F}_2^n\}\ .$$

Indeed, it is necessary for G to be well-defined that the left-hand side of the output of $x \mapsto \mathcal{A}(x, F(x))$ is a permutation. As a consequence, only a few permutations $\mathcal{A}$ yield valid functions G. The following definition captures this intuition.

Definition 7 (Admissible affine permutations)**.** *Let F be a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$. We say that the affine permutation $\mathcal{A}$ of $\mathbb{F}_2^n \times \mathbb{F}_2^m$ is* admissible *for F if we can define a function G such that*

$$\mathcal{A}\left(\{(x, F(x)), x \in \mathbb{F}_2^n\}\right) = \{(x, G(x)), x \in \mathbb{F}_2^n\}\,.$$

3 Recovering EA-equivalence for Quadratic Functions

3.1 The Jacobian Matrix

Notation 8. *In the sequel, the canonical basis of $\mathbb{F}_2^n$ is denoted as $(e_1, \ldots, e_n)$.*

Definition 9. *Let F be a function from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$. The* Jacobian *of F at $x \in \mathbb{F}_2^n$ is the parameterised matrix defined by*

$$\operatorname{Jac} F(x) \;:=\; \begin{pmatrix} \Delta_{e_1} F_1(x) & \cdots & \Delta_{e_n} F_1(x) \\ \vdots & & \vdots \\ \Delta_{e_1} F_m(x) & \cdots & \Delta_{e_n} F_m(x) \end{pmatrix}. \tag{1}$$

On the other hand, given an m–tuple of polynomials $P = (P_1, \ldots, P_m) \in \mathbb{F}_2[X_1, \ldots, X_n]^m$, we define the Jacobian matrix of P as

$$\mathscr{J}P(x) \;:=\; \begin{pmatrix} \frac{\partial P_1}{\partial x_1} & \cdots & \frac{\partial P_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial P_m}{\partial x_1} & \cdots & \frac{\partial P_m}{\partial x_n} \end{pmatrix} \in \mathbb{F}_2[X_1, \ldots, X_n]^{m \times n}.$$

Remark 10. The two notions are strongly related to each other. In particular, for a Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, denote by $P_F \in \mathbb{F}_2[X_1, \ldots, X_n]^m$ the polynomial representation of F **in algebraic normal form**, then

$$\operatorname{Jac} F(x) = \mathscr{J}P_F(x).$$

This equality can be easily checked on monomials and then extended by linearity.

Note however the importance of being in algebraic normal form: for instance in one variable, if $P(x) = x^2$, then $\frac{\partial P}{\partial x} = 0$, while the algebraic normal form of P is x whose derivative is 1.

3.2 The Jacobian Matrices of EA-equivalent Functions

Suppose that $G = A \circ F \circ B + C$, for some affine permutations A and B and an affine function C, respectively defined as

$$\forall x \in \mathbb{F}_2^n, \quad A(x) = A_0 x + a, \quad B(x) = B_0 x + b, \quad \text{and} \quad C(x) = C_0 x + c$$

where A_0, B_0 are non-singular matrices in $\mathbb{F}_2^{n \times n}$ and $\mathbb{F}_2^{m \times m}$ respectively, $a, c \in \mathbb{F}_2^m$ and $b \in \mathbb{F}_2^n$. Note that, after replacing a by $a + c$, one can suppose that $c = 0$ and hence that C is linear. We always proceed this way in the sequel.

Denote by P_F, P_G some polynomial representations of F, G. Then we also have

$$P_G = P_A \circ P_F \circ P_B + P_C.$$

Then, considering Jacobian matrices, one can apply the well–known *chain rule formula for functions of several variables*, *i.e.* the formula for the Jacobian of compositions of functions, namely:

$$\mathscr{J}P_G(x) = A_0 \cdot \mathscr{J}P_F(B(x)) \cdot B_0 + C_0 \tag{2}$$

Indeed, the respective Jacobians of A, B and C are A_0, B_0 and C_0. Unfortunately, this chain rule formula does not extend to Jacobians of Boolean functions because, as already observed in Remark 10, the operations of derivation and of reduction to the algebraic normal form do not commute. To clarify this issue, let us consider an elementary example.

Example 11. Let $P_F(x_1, x_2) = x_1 x_2$ be a polynomial representing a function $F : \mathbb{F}_2^2 \to \mathbb{F}_2$. Consider the affine map $B : (x_1, x_2) \mapsto (x_1 + x_2, x_2)$. Set $P_G = P_F \circ B = x_1 x_2 + x_2^2$ whose algebraic normal form is $G = x_1 x_2 + x_2$. Now, the Jacobian matrices of F and G are

$$\operatorname{Jac} F(x) = (x_2 \;\; x_1) \quad \text{and} \quad \operatorname{Jac} G(x) = (x_2 \;\; x_1 + 1)$$

and

$$\operatorname{Jac} F(B(x)) \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = (x_2 \quad x_1 + x_2) \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = (x_2 \ x_1),$$

which differs from $\operatorname{Jac} G(x)$. On the other hand, if we consider polynomials instead of Boolean functions, we have

$$\mathscr{J} P_F(B(x)) \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = (x_2 \ x_1) = \mathscr{J} G(x).$$

In the sequel, we prove that, while we observed that the chain rule formula in several variables is false in general, in the context of quadratic functions it is possible to get a very similar formula using the so-called *linear part* of the Jacobian. This will be the crux of our algorithm to follow.

3.3 The Jacobian Matrix of a Quadratic Function

From now on, we suppose that F is quadratic, i.e. its algebraic normal form has degree 2. In this case, the entries of the associated Jacobian matrix, in the sense of Definition 9, are polynomials of degree 1 and we will focus on their homogeneous parts.

Definition 12. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a quadratic function. We denote by $\operatorname{Jac}_{\text{lin}} F(x)$ the linear part of $\operatorname{Jac} F(x)$, i.e. the matrix whose entries are the homogeneous parts of degree 1 of the entries of $\operatorname{Jac} F(x)$:*

$$\forall x \in \mathbb{F}_2^n, \quad \operatorname{Jac} F(x) = \operatorname{Jac}_{\text{lin}} F(x) + \operatorname{Jac} F(0) .$$

Equivalently, $\operatorname{Jac}_{\text{lin}} F(x) = (J_{i,j}(x))_{i,j}$ with

$$J_{i,j}(x) = \Delta_{e_j} F_i(x) + \Delta_{e_j} F_i(0), \tag{3}$$

where $(e_1, \ldots, e_n)$ denote the canonical basis of $\mathbb{F}_2^n$ (Notation 8).

It is worth noticing that the linear part of the Jacobian of a quadratic function corresponds to the coefficients of the quadratic monomials in the algebraic normal forms of the coordinates of F. Equivalently, $\operatorname{Jac}_{\text{lin}} F(x)$ is the Jacobian of the degree-2 homogeneous part of the algebraic normal form of F as explained by the following statement.

Proposition 13. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a quadratic function. Let*

$$F_i(x_1, \ldots, x_n) = \sum_{k<\ell} Q^i_{k,\ell} x_k x_\ell + \sum_{k=1}^{n} c^i_k x_k + \varepsilon^i$$

denote the algebraic normal form of the i-th coordinate of F, $1 \leqslant i \leqslant m$, where all coefficients $Q^i_{k,\ell}, c^i_k, \varepsilon^i$ lie in $\mathbb{F}_2$ and $Q^i_{k,\ell} = 0$ when $k \geqslant \ell$. Then, the entries $J_{i,j}(x)$ of $\operatorname{Jac}_{\text{lin}} F(x)$ are

$$J_{i,j}(x) = \sum_{k=1}^{n} (Q^i_{k,j} + Q^i_{j,k}) x_k, \ 1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n .$$

Proof. For any i and j, $1 \leqslant i \leqslant m$ and $1 \leqslant j \leqslant n$, we have

$$\Delta_{e_j} F_i(x) = \sum_{k<j} Q^i_{k,j} x_k + \sum_{k>j} Q^i_{j,k} x_k + c_j .$$

We then deduce from (3) that

$$J_{i,j}(x) = \Delta_{e_j} F_i(x) + \Delta_{e_j} F_i(0) = \sum_{k=1}^{n} (Q^i_{k,j} + Q^i_{j,k}) x_k .$$

□

The linear part of the Jacobian of a quadratic function is a useful mathematical object since the values of all derivatives of the function can be derived from this matrix, as shown in the following proposition.

Proposition 14. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a quadratic function and let $\mathrm{Jac}_{\mathrm{lin}}\, F(x) = (J_{i,j}(x))_{\substack{1\leqslant i\leqslant m\\ 1\leqslant j\leqslant n}}$ denote the linear part of its Jacobian. Then, for any $a = (a_1, \dots, a_n) \in \mathbb{F}_2^n$, any $i \in \{1, \dots, m\}$ and any $x \in \mathbb{F}_2^n$, we have:*

$$\Delta_a F_i(x) + \Delta_a F_i(0) = \sum_{j=1}^{n} a_j J_{i,j}(x) \ .$$

Hence, for any $a \in \mathbb{F}_2^n$ and any $x \in \mathbb{F}_2^n$, we have:

$$\Delta_a F(x) + \Delta_a F(0) = \mathrm{Jac}_{\mathrm{lin}}\, F(x) \cdot a \ .$$

Proof. We proceed by induction on the Hamming weight of a. The statement obviously holds when $wt(a) = 1$ since it corresponds to Equation (3). Suppose now that it holds for all words of weight at most $(w - 1)$ and consider a word a of weight $w \leqslant n$. Let $\ell \in \{1, \dots, n\}$ be such that $a_\ell = 1$, and $a' = a + e_\ell$. Then,

$$\begin{aligned}
\Delta_a F(x) &= F(x + a' + e_\ell) + F(x) \\
&= F(x + a' + e_\ell) + F(x + a') + F(x + a') + F(x) \\
&= \Delta_{e_\ell} F(x + a') + \Delta_{a'} F(x) \ .
\end{aligned}$$

Then, we have

$$\begin{aligned}
\Delta_a F(x) + \Delta_a F(0) &= \Delta_{e_\ell} F(x + a') + \Delta_{e_\ell} F(a') + \Delta_{a'} F(x) + \Delta_{a'} F(0) \\
&= \Delta_{e_\ell} F(x) + \Delta_{e_\ell} F(0) + \Delta_{a'} F(x) + \Delta_{a'} F(0)
\end{aligned}$$

where the last equality is deduced from the fact that $\Delta_{e_\ell} F(x)$ is affine. We deduce from the induction hypothesis that, for any i,

$$\begin{aligned}
\Delta_a F_i(x) + \Delta_a F_i(0) &= \Delta_{e_\ell} F_i(x) + \Delta_{e_\ell} F_i(0) + \sum_{j=1}^{n} a'_j J_{i,j}(x) \\
&= \sum_{j=1}^{n} a_j J_{i,j}(x) \ .
\end{aligned}$$

□

As an immediate corollary which will be extensively used later, it appears that the roles of a and x in the previous proposition can be switched.

Corollary 15. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a quadratic function. Then, for any $a, x \in \mathbb{F}_2^n$, we have*

$$\mathrm{Jac}_{\mathrm{lin}}\, F(x) \cdot a = \mathrm{Jac}_{\mathrm{lin}}\, F(a) \cdot x \ .$$

Proof. From Proposition 14, we have that, for any $a, x \in \mathbb{F}_2^n$,

$$\mathrm{Jac}_{\mathrm{lin}}\, F(x) \cdot a = \Delta_a F(x) + \Delta_a F(0) = \Delta_x \Delta_a F(0) = \Delta_a \Delta_x F(0) = \mathrm{Jac}_{\mathrm{lin}}\, F(a) \cdot x \ .$$

□

Corollary 16. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a quadratic function. Then, for any $a \in \mathbb{F}_2^n$, we have*

$$\mathrm{Jac}_{\mathrm{lin}}\, F(a) \cdot a = 0.$$

Proof. Let $i \in \{1, \ldots, m\}$. From Proposition 14,

$$\mathrm{Jac}_{\mathrm{lin}}\, F(a) \cdot a = \Delta_a F(a) + \Delta_a F(0) = F(a + a) + F(a) + F(a) + F(0) = 0.$$

□

Using Proposition 14, we can exhibit the relation between the linear parts of the Jacobians of two EA-equivalent quadratic functions. This relation is very close to the chain rule formula in differential calculus. In addition it will be of particular interest for recovering the triple of functions (A, B, C) such that $G = A \circ F \circ B + C$ because it does not involve C.

Proposition 17. *Let F and G be two EA-equivalent quadratic functions from $\mathbb{F}_2^n$ into $\mathbb{F}_2^m$ with $G = A \circ F \circ B + C$ for some affine permutations A and B, and some affine function C. Then,*

$$\forall x \in \mathbb{F}_2^n, \quad \mathrm{Jac}_{\mathrm{lin}}\, G(x) = A_0 \cdot \mathrm{Jac}_{\mathrm{lin}}\, F(B(x)) \cdot B_0 \ , \tag{4}$$

where A_0 and B_0 denote the matrices corresponding to the linear parts of A and B.

Proof. Let A_0, B_0 and C_0 denote the matrices corresponding to the linear parts of A,B and C. Then,

$$\begin{aligned} \Delta_{e_j} G(x) &= A \circ F \circ B(x + e_j) + A \circ F \circ B(x) + C(x + e_j) + C(x) \\ &= A_0 \left[F\left(B(x) + B_0 e_j\right) + F\left(B(x)\right)\right] + C_0 e_j \\ &= A_0 \Delta_{B_0 e_j} F\left(B(x)\right) + C_0 e_j \ . \end{aligned}$$

It follows that

$$\begin{aligned} \Delta_{e_j} G(x) + \Delta_{e_j} G(0) &= A_0 \Delta_{B_0 e_j} F\left(B(x)\right) + C_0 e_j + A_0 \Delta_{B_0 e_j} F\left(B(0)\right) + C_0 e_j \\ &= A_0 \left[\Delta_{B_0 e_j} F\left(B(x)\right) + \Delta_{B_0 e_j} F\left(B(0)\right)\right] \ . \end{aligned}$$

Let now $\mathrm{Jac}_{\mathrm{lin}}\ F(x) = (J_{i,j}(x))_{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant n}}$, so that Proposition 14 implies the following for any i satisfying $1 \leqslant i \leqslant m$:

$$\begin{aligned} \Delta_{B_0 e_j} F_i\left(B(x)\right) + \Delta_{B_0 e_j} F_i\left(B(0)\right) &= \sum_{k=1}^{n} (B_0 e_j)_k J_{i,k}(B(x)) \\ &= \sum_{k=1}^{n} (B_0)_{k,j} J_{i,k}(B(x)) = \left[\mathrm{Jac}_{\mathrm{lin}}\, F(B(x)) B_0\right]_{i,j} \ . \end{aligned}$$

Combining the last two equalities, we obtain:

$$\left[\mathrm{Jac}_{\mathrm{lin}}\, G(x)\right]_{i,j} = \Delta_{e_j} G_i(x) + \Delta_{e_j} G_i(0) = \left[A_0 \mathrm{Jac}_{\mathrm{lin}}\, F(B(x)) B_0\right]_{i,j} \ .$$

□

It is worth noticing that, when $m = n$, the linear part of the Jacobian of a quadratic homogeneous function is related to a special class of symmetric matrices over $\mathbb{F}_{2^n}$ called QAM and introduced in [YWL14]. This notion of QAM arises by exhibiting a one-to-one correspondence between quadratic homogeneous functions over $\mathbb{F}_2^n$ and symmetric matrices over $\mathbb{F}_{2^n}$ with diagonal elements equal to zero [YWL14]. This correspondence is detailed in the following definition.

Definition 18 ([YWL14]). *Let $\mathcal{B} = (\alpha_1, \alpha_2, \dots, \alpha_n)$ be a basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$ and $\varphi : (x_1, \dots, x_n) \in \mathbb{F}_2^n \mapsto \sum_{i=1}^n x_i\alpha_i \in \mathbb{F}_{2^n}$. Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic homogeneous function and $P(X) = \sum_{k<\ell} q_{k,\ell} X^{2^{k-1}+2^{\ell-1}}$ be the quadratic homogeneous polynomial in $\mathbb{F}_{2^n}[X]$ such that*

$$F(x) = \varphi^{-1} \circ P \circ \varphi(x),\ \forall x \in \mathbb{F}_2^n\ .$$

Then the matrix associated with F with respect to $\mathcal{B}$ *is*

$$H = M^T C_P M$$

where C_P is the $n \times n$ symmetric matrix over $\mathbb{F}_{2^n}$ defined by $(C_P)_{k,\ell} = (C_P)_{\ell,k} = q_{k,\ell}$ for all $1 \leqslant k < \ell \leqslant n$, $(C_P)_{k,k} = 0$, and M is the Moore matrix associated to $\mathcal{B}$, i.e., $M_{i,j} = \alpha_j^{2^{i-1}}$.

The advantage of this construction is that for a quadratic homogeneous function, being APN can be characterized by some algebraic properties of the associated matrix [YWL14]. We now show that the matrix $H = M^T C_P M$ is related to the linear part of the Jacobian of F.

Proposition 19. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic homogeneous function and H be the matrix associated with F with respect to the basis $\mathcal{B} = (\alpha_1, \alpha_2, \dots, \alpha_n)$ in the sense of Definition 18. Then, for all $x \in \mathbb{F}_2^n$,*

$$\mathrm{Jac}_{\mathrm{lin}}\, F(x) \cdot \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} = (x_1, \dots, x_n) \cdot H\ .$$

Proof. Theorem 1 in [YWL14] shows that, for all j, $1 \leqslant j \leqslant n$,

$$\Delta_{e_j}(\varphi \circ F)(x) = xH_j$$

where H_j denotes Column j of H and $\varphi : x \in \mathbb{F}_2^n \mapsto (\sum_{i=1}^n x_i\alpha_i) \in \mathbb{F}_{2^n}$. Using that F is homogeneous, it follows that, for all (i, j),

$$[\mathrm{Jac}_{\mathrm{lin}}\, F(x)]_{i,j} = \Delta_{e_j} F_i(x)\ .$$

It implies that

$$\sum_{i=1}^n \alpha_i [\mathrm{Jac}_{\mathrm{lin}}\, F(x)]_{i,j} = \Delta_{e_j}(\varphi \circ F)(x) = xH_j\ ,$$

and the result directly follows. □

In other words, $\mathrm{Jac}_{\mathrm{lin}}\, F(x)$ is three-dimensional in nature since all entries in the matrix are n-variable linear functions. The previous proposition shows that Matrix H defined in [YWL14] is another way to represent the same object with a 2-dimensional structure by using $\mathbb{F}_{2^n}$ as a coefficient field. Characterizing the fact that F is APN from the properties of H can then be reformulated in terms of Jacobians as we will prove in Section 3.4.2. However, our result is more general since it applies even if F is not homogeneous and also to functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ with $m \neq n$. More importantly, the representation in terms of Jacobians is more convenient for analyzing EA-equivalence.

3.4 Solving the EA-equivalence Problem for Quadratic Functions

Our algorithm takes as input two quadratic functions $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and returns, if it exists, a triple (A, B, C) of affine functions such that A, B are permutations and $G = A \circ F \circ B + C$. Denote by

$$A(x) = A_0 x + a \quad B(x) = B_0 x + b \quad \text{and} \quad C(x) = C_0 x + c,$$

where $A_0 \in \mathbb{F}_2^{m\times m}$, $B_0 \in \mathbb{F}_2^{n\times n}$, $C_0 \in \mathbb{F}_2^{m\times n}$, $a \in \mathbb{F}_2^m$, $b \in \mathbb{F}_2^n$ and $c \in \mathbb{F}_2^m$.

As already noted before, replacing a by $a + c$ one can suppose that $c = 0$ and hence that C is linear. In addition, since the functions are quadratic, one may also suppose that $b = 0$. Indeed, it suffices to observe that

$$\forall x \in \mathbb{F}_2^n, \quad F(B_0 x + b) = F(B_0 x) + \Delta_b F(B(x))$$

and, since F is quadratic, then $\Delta_b F(B(x))$ is affine and its linear part equals $\Delta_b F(B_0 x) + \Delta_b F(0)$. Therefore, replacing C by $x \mapsto C(x) + \Delta_b F(B_0 x) + \Delta_b F(0)$ and a by $a + \Delta_b F(0)$, one can suppose that both B and C are linear.

In summary, our objective is to find, if it exists, a 4–tuple (A_0, B_0, C_0, a) such that A_0, B_0 are non-singular, and

$$\forall x \in \mathbb{F}_2^n, \quad G(x) = A_0 \cdot F(B_0 x) + C_0 x + a \ .$$

The key of our algorithm rests on Proposition 17 which asserts that

$$\forall x \in \mathbb{F}_2^n, \quad \mathrm{Jac}_{\mathrm{lin}}\, G(x) = A_0 \cdot \mathrm{Jac}_{\mathrm{lin}}\, F(B_0 x) \cdot B_0 \ . \tag{5}$$

This permits first to search for the pair (A_0, B_0), and then, once it is computed, to deduce the remainder of the 4–tuple. The search for this pair (A_0, B_0) relies on two main ideas:

(i) If a pair (v, w) is known to satisfy $B_0 v = w$, then the pair (A_0^{-1}, B_0) is a solution of the affine system with unknowns $(X, Y) \in \mathbb{F}_2^{m\times m} \times \mathbb{F}_2^{n\times n}$:

$$\left\{ \begin{array}{rcl} X \cdot \mathrm{Jac}_{\mathrm{lin}}\, G(v) - \mathrm{Jac}_{\mathrm{lin}}\, F(w) \cdot Y & = & 0 \\ Y \cdot v & = & w. \end{array} \right. \tag{6}$$

(ii) Since A_0, B_0 are non-singular, then, according to (5), for any $x \in \mathbb{F}_2^n$, the matrices $\mathrm{Jac}_{\mathrm{lin}}\, G(x)$ and $\mathrm{Jac}_{\mathrm{lin}}\, F(B_0 x)$ have the same rank.

3.4.1 Sketch of the Algorithm

The search for the pair (A_0, B_0) will be done by trying to guess pairs $(v_1, w_1), \ldots, (v_s, w_s)$ of elements of $\mathbb{F}_2^n \times \mathbb{F}_2^n$ such that for any i, $B_0 v_i = w_i$. For each such guess, we solve a concatenation of systems of the form (6) and check whether it leads to a relevant solution. If not, we try with another guess.

Therefore, the complexity analysis of the algorithm is directly related to the average number of guesses we will have to do, which should be the smallest possible. This motivates the following studies.

- Using (ii), any guess (v_i, w_i) should be chosen so that $\mathsf{rank}\mathrm{Jac}_{\mathrm{lin}}\, G(v_i) = \mathsf{rank}\mathrm{Jac}_{\mathrm{lin}}\, F(w_i)$. Therefore, the search is much easier if we seek elements $v \in \mathbb{F}_2^n$ (resp. $w \in \mathbb{F}_2^n$) such that $\mathsf{rank}\mathrm{Jac}_{\mathrm{lin}}\, G(v)$ (resp. $\mathsf{rank}\mathrm{Jac}_{\mathrm{lin}}\, F(w)$) occurs rarely in the rank table of $\mathrm{Jac}_{\mathrm{lin}}\, G(x)$ (resp. $\mathrm{Jac}_{\mathrm{lin}}\, F(x)$). This motivates the study of this rank table in Section 3.4.2.

- If the number s of simultaneous guesses $(v_1, w_1), \ldots, (v_s, w_s)$ should be the smallest possible, it should also be large enough so that the linear system:

$$\left\{ \begin{array}{rcl} X \cdot \mathrm{Jac}_{\mathrm{lin}}\, G(v_i) - \mathrm{Jac}_{\mathrm{lin}}\, F(w_i) \cdot Y & = & 0 \\ Y \cdot v_i & = & w_i \end{array} \right. \qquad \forall i \in \{1, \ldots, s\}, \tag{7}$$

has a unique solution or a "small enough" affine space of solutions. Thus, the rank of such a system, which is nothing but a concatenation of s systems of the form (6), is investigated in Section 3.4.3.

- As soon as a possibly valid pair (A_0, B_0) is found, there remains to recover C_0, a. This boils down to linear algebra and is detailed in Section 3.4.4.

3.4.2 Rank Table and Connection with the Differential Spectrum

As explained earlier, a part of the algorithm consists in guessing an s–tuple of vectors in $\mathbb{F}_2^n$, which *a priori* requires $\mathrm{O}\left(2^{sn}\right)$ trials. This number of trials can be drastically improved using the *rank tables.*

Definition 20. *The* rank table $\mathcal{R}(F)$ *of* F *is a table with* $(\min(m, n) + 1)$ *entries indexed by* $\{0, \ldots, \min(m, n)\}$ *and*

$$\forall j \in \{0, \ldots, \min(m, n)\}, \quad \mathcal{R}(F)[j] := \{x \in \mathbb{F}_2^n \mid \mathsf{rank}(\mathrm{Jac}_{\mathrm{lin}}\, F(x)) = j\}.$$

The rank distribution $\mathcal{R}_{\mathrm{dist}}(F)$ *of* F *is defined as*

$$\forall j \in \{0, \ldots, \min(m, n)\}, \quad \mathcal{R}_{\mathrm{dist}}(F)[j] := \#\mathcal{R}(F)[j].$$

Lemma 21. *The computation of the rank table can be performed in* $\mathrm{O}\left(\max(n, m)^{\omega} 2^n\right)$ *operations.*

Remark 22. It is worth noticing that we denote as *rank table* an object which strongly differs from that used by Dinur in [Din18], where another "rank table" is used to decide (non-extended) affine equivalence. Dinur's "rank table" consists in considering the *symbolic ranks* of F, G [Din18, Section 2, page 418], which are completely different objects.

We now show that there is a one-to-one correspondence between the rank distribution of a quadratic function and the distribution of its differential spectrum.

Proposition 23. *Let* $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ *be a quadratic function and* $\mathcal{R}_{\mathrm{dist}}(F)$ *be its rank distribution. Let* $\mathcal{D}_{\mathrm{dist}}(F)$ *denote the distribution of the differential spectrum of* F*, i.e.,*

$$\mathcal{D}_{\mathrm{dist}}(F)[k] = \#\left\{(a, b) \in (\mathbb{F}_2^n)^2 \mid \delta_F(a, b) = k\right\} .$$

Then, for any r*,* $0 \leqslant r \leqslant \min(m, n)$*,*

$$\mathcal{R}_{\mathrm{dist}}(F)[r] = 2^{-r} \mathcal{D}_{\mathrm{dist}}(F)[2^{n-r}] .$$

Proof. Let $a \in \mathbb{F}_2^n$. Proposition 14 and Corollary 15 imply that

$$\mathrm{Jac}_{\mathrm{lin}}\, F(a) \cdot x = \mathrm{Jac}_{\mathrm{lin}}\, F(x) \cdot a = \Delta_a F(x) + \Delta_a F(0) .$$

Then, the elements $x \in \mathbb{F}_2^n$ in the right kernel of $\mathrm{Jac}_{\mathrm{lin}}\, F(a)$ are those such that $\Delta_a F(x) + \Delta_a F(0) = 0$. Since F is quadratic, this second set is a linear space whose dimension equals i where

$$\delta_F(a, b) \in \{0, 2^i\}, \ \forall b \in \mathbb{F}_2^n .$$

Let r denote the rank of $\mathrm{Jac}_{\mathrm{lin}}\, F(a)$. We then deduce that

$$\begin{aligned}
\#\{x \in \mathbb{F}_2^n : \mathrm{Jac}_{\mathrm{lin}}\, F(a) \cdot x = 0\} &= 2^{n-r} \\
&= \#\{x \in \mathbb{F}_2^n : \Delta_a F(x) + \Delta_a F(0) = 0\} \\
&= 2^i \ ,
\end{aligned}$$

where $\{\delta_F(a,b), b \in \mathbb{F}_2^m\} = \{0, 2^i\}$. Then, the entries in the row of the DDT defined by a are 0 and 2^{n-r}, and the value 2^{n-r} appears 2^r times. It follows that

$$\mathcal{R}_{\mathrm{dist}}(F)[r] = 2^{-r}\mathcal{D}_{\mathrm{dist}}(F)[2^{n-r}] \ .$$

□

Remark 24. It is worth noticing that the previous proposition implies that, for any quadratic function F, we have $\mathcal{R}_{\mathrm{dist}}(F)[n] = 0$, since the values in the differential spectrum of a Boolean function are always even.

As another consequence, we get that the differential spectrum of a quadratic function contains two values only, 0 and $\delta(F)$, if and only if the matrices $\mathrm{Jac}_{\mathrm{lin}}\, F(x)$ for all nonzero $x \in \mathbb{F}_2^n$ have the same rank. This includes for instance the case of bent functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$, $n \geqslant 2m$, and the case of APN functions.

Corollary 25. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a quadratic function . Then, F is APN if and only if $\mathrm{Jac}_{\mathrm{lin}}\, F(x)$ has rank $(n-1)$ for all nonzero x.*

3.4.3 On the Rank of the Linear System (6)

Proposition 26. *Let F, G be two quadratic functions and $v, w \in \mathbb{F}_2^n$ such that*

$$\mathsf{rank}\mathrm{Jac}_{\mathrm{lin}}\, G(v) = \mathsf{rank}\mathrm{Jac}_{\mathrm{lin}}\, F(w).$$

Denote by r the above rank. Then the linear part of the system (6), i.e. *the linear system with unknowns $(X,Y) \in \mathbb{F}_2^{m\times m} \times \mathbb{F}_2^{n\times n}$:*

$$X \cdot \mathrm{Jac}_{\mathrm{lin}}\, G(v) - \mathrm{Jac}_{\mathrm{lin}}\, F(w) \cdot Y = 0 \tag{8}$$

has $m^2 + n^2$ unknowns, mn equations and rank less than or equal to $r(m+n-r)$. Moreover, the affine system (6) has m^2+n^2 unknowns, $(m+1)n$ equations and rank less than or equal to $r(m+n-r) + (n-r)$.

Proof. The number of unknowns corresponds to the number of entries of X, Y. The number of linear equations equals the number of entries of the resulting matrix $X \cdot \mathrm{Jac}_{\mathrm{lin}}\, G(v) - \mathrm{Jac}_{\mathrm{lin}}\, F(w) \cdot Y$ which is mn. Let us investigate the rank. Since $\mathrm{Jac}_{\mathrm{lin}}\, G(v)$ has rank r, there exists a non-singular matrix $U \in \mathbb{F}_2^{n\times n}$ such that the $n-r$ rightmost columns of $\mathrm{Jac}_{\mathrm{lin}}\, G(v) \cdot U$ are zero. Hence it has the following shape:

$$\mathrm{Jac}_{\mathrm{lin}}\, G(v) \cdot U = \begin{array}{c} \begin{pmatrix} \mathrm{J}_1 G & (0) \\ \mathrm{J}_2 G & (0) \end{pmatrix} \begin{array}{l} \updownarrow r \\ \updownarrow m-r \end{array} \\ \begin{array}{cc} \underleftrightarrow{\;r\;} & \underleftrightarrow{n-r} \end{array} \end{array}$$

for some matrices $\mathrm{J}_1 G, \mathrm{J}_2 G$. Similarly, there exists a non-singular matrix $V \in \mathbb{F}_2^{m\times m}$ such that $V \cdot \mathrm{Jac}_{\mathrm{lin}}\, F(w)$ has the following shape:

$$V \cdot \mathrm{Jac_{lin}}\, F(w) = \begin{array}{cc} \begin{pmatrix} \mathrm{J}_1F & \mathrm{J}_2F \\ (0) & (0) \end{pmatrix} & \begin{array}{l} \updownarrow r \\ \updownarrow m-r \end{array} \\ \begin{array}{cc} \xleftrightarrow[r]{} & \xleftrightarrow[n-r]{} \end{array} & \end{array}$$

for some matrices $\mathrm{J}_1F, \mathrm{J}_2F$. Thus, setting $X' := VX$ and $Y' := YU$, we get a new and equivalent linear system

$$X' \cdot \mathrm{Jac_{lin}}\, G(v) \cdot U - V \cdot \mathrm{Jac_{lin}}\, F(v) \cdot Y' = 0. \tag{9}$$

If we denote the block decompositions of X', Y' as

$$X' = \begin{array}{cc} \begin{pmatrix} X_1' & X_2' \\ X_3' & X_4' \end{pmatrix} & \begin{array}{l} \updownarrow r \\ \updownarrow m-r \end{array} \\ \begin{array}{cc} \xleftrightarrow[r]{} & \xleftrightarrow[m-r]{} \end{array} & \end{array} \quad \text{and} \quad Y' = \begin{array}{cc} \begin{pmatrix} Y_1' & Y_2' \\ Y_3' & Y_4' \end{pmatrix} & \begin{array}{l} \updownarrow r \\ \updownarrow n-r \end{array} \\ \begin{array}{cc} \xleftrightarrow[r]{} & \xleftrightarrow[n-r]{} \end{array} & \end{array}$$

Then, the block decomposition of the system (9) gives:

$$\begin{array}{cc} \begin{array}{r} r \updownarrow \\ m-r \updownarrow \end{array} & \begin{pmatrix} X_1'\mathrm{J}_1G + X_2'\mathrm{J}_2G + \mathrm{J}_1FY_1' + \mathrm{J}_2FY_3' & \mathrm{J}_1FY_2' + \mathrm{J}_2FY_4' \\ X_3'\mathrm{J}_1G + X_4'\mathrm{J}_2G & (0) \end{pmatrix} \\ & \begin{array}{cc} \xleftrightarrow[r]{\qquad\qquad\qquad\qquad} & \xleftrightarrow[n-r]{\qquad} \end{array} \end{array} = \begin{array}{cc} \begin{pmatrix} (0) & (0) \\ (0) & (0) \end{pmatrix} & \begin{array}{l} \updownarrow r \\ \updownarrow m-r \end{array} \\ \begin{array}{cc} \xleftrightarrow[r]{} & \xleftrightarrow[n-r]{} \end{array} & \end{array}$$

Each of the mn entries of the left-hand matrix yields a linear equation relating the entries of X', Y'. Hence, one can forget the entries of the bottom right–hand corner yielding "$0 = 0$" equations, which leaves $r(m+n-r)$ equations. This yields the upper bound on the rank of System (9) and hence that of System (8).

Concerning the whole System (6), including the affine equations $Y \cdot v = w$ consists in joining n additional affine equations. However, these equations are never independent. Indeed, if (X, Y) is solution of (8), then $X\mathrm{Jac_{lin}}\, G(v) = \mathrm{Jac_{lin}}\, F(w)Y$ and, according to Corollary 16, we deduce that $Y \cdot v$ should lie in the right kernel of $\mathrm{Jac_{lin}}\, F(w)$ which has dimension $n-r$. Thus, the n additional affine equations on the entries of Y given by $Y \cdot v = w$ impose at most $n - r$ new independent conditions on the entries of Y. □

Our experimental observations show that for a fixed pair (v, w) the upper bound on the rank of (6) is sharp. On the other hand, when considering s–tuples of pairs $(v_1, w_1), \ldots, (v_s, w_s)$ and solving the concatenation of s systems of the form (6), *i.e.* a system of the form (7), does not in general lead to a system of rank

$$\sum_{i=1}^{s} \left(r_i(m+n-r_i) + (n-r_i) \right) \ , \tag{10}$$

but to a slightly smaller rank and we did not succeed in getting a sharper estimate as soon as $s > 1$, which is actually the use case of our algorithm to follow.

Example 27. Using SageMath [S+19], we computed the rank of the systems (8) (without the affine equations) and (6) (including affine equations) for various values of m, n. We also evaluated the rank of Systems (7), *i.e.* the concatenation of s distinct systems of the form (6). Results are presented in Table 2. These observations show that as soon as $s > 1$, the upper bound (10) is not sharp enough. On the other hand, our experimental observations show that for many use cases, the choice $s = 3$ is relevant in order to have a system with few solutions.

For APN functions, the smallest rank occurring in the rank distribution is $n - 1$. In this situation the rank of a system of the form (8) is larger and in such a case, the choice $s = 2$ might be sufficient, as suggested by the following example.

m	n	m^2+n^2	s	Jacobian ranks	Expected ranks for (8) and (6)	Observed ranks (intervals)
6	6	72	1	(3)	(27, 30)	(27, 30)
6	6	72	1	(4)	(32, 34)	(32, 34)
6	8	100	1	(4)	(40, 44)	(40, 44)
8	6	100	1	(4)	(40, 42)	(40, 42)
6	6	72	2	(3,3)	(54, 60)	(48...53, 50...54)
6	6	72	2	(3,4)	(59, 64)	(52...56, 56...57)
6	6	72	2	(4,4)	(64, 68)	(56...60, 60...61)
6	8	100	2	(4,4)	(80, 88)	(72...80, 78...82)
8	6	100	2	(4,4)	(80, 84)	(68...72, 72...73)
6	6	72	3	(3,4,4)	(72, 72)	(65...72, 69...72)
6	6	72	3	(4,4,4)	(72, 72)	(60...72, 66...72)
6	8	100	3	(4,4,4)	(100, 100)	(90...100, 93...100)
8	6	100	3	(4,4,5)	(100, 100)	(95...96, 96)

Table 2: This table summarizes some experimental results. The column *Jacobian ranks* gives the ranks of $\mathrm{Jac}_{\mathrm{lin}}\, G(v_i)$ (or equivalently $\mathrm{Jac}_{\mathrm{lin}}\, F(w_i)$) for the (v_i, w_i)'s we tested.

Example 28. Consider the function

$$\left\{ \begin{array}{ccc} \mathbb{F}_{2^n} & \to & \mathbb{F}_{2^n} \\ x & \mapsto & x^{2^i+1} \end{array} \right.$$

with i prime to n. Such a function is known to be APN and, when regarded as a function $\mathbb{F}_2^n \to \mathbb{F}_2^n$ it is quadratic. For $n = 5$ and $i = 1$, the choice $s = 2$ provides a linear system whose solution space has dimension 5, while $s = 3$ provides a solution space of dimension $\leqslant 1$. For $n = 7$, choosing $s = 2$ yields solutions spaces of dimension 7 while $s = 3$ these spaces have dimension $\leqslant 1$.

Therefore, for such functions, it seems unclear which choice for s is the most relevant. On one hand, choosing $s = 2$ will imply a brute-force search over a set of size $\mathrm{O}\left(2^{2n}\right)$ but each step will require a second brute-force in the solution space of System (7), which is reasonable for $n = 5, 7$ but non negligible. On the other hand, the brute-force search is performed on a set of guesses of $\mathrm{O}\left(2^{3n}\right)$ elements, but each step of the search is much less expensive.

3.4.4 Deducing the full extended-affine equivalence

Once a pair (A_0, B_0) is computed, the remainder of the 4–tuple can be computed as follows. Let G_1 to be the function defined by $\forall x \in \mathbb{F}_2^n,\ G_1(x) := A_0 F(B_0 x)$ and recall that $G(x) = A_0 F(B_0 x) + C_0 x + a$. Since G_1 can be computed from the triple (F, A_0, B_0), then, one gets a using

$$a = G(0) + G_1(0).$$

Finally, C_0 can be computed as the linear map satisfying

$$\forall x \in \mathbb{F}_2^n, \quad C_0 x = G(x) + G_1(x) + a.$$

3.4.5 The Algorithm

The pseudo-code of the full algorithm for recovering extended-affine equivalence is summarized in Algorithm 1. Below we give a description of the algorithm with several comments.

1. Compute the rank tables and the rank distributions of $\mathrm{Jac}_{\mathrm{lin}}\, F(x)$ and $\mathrm{Jac}_{\mathrm{lin}}\, G(x)$. If these distributions differ, then the functions are not equivalent.

2. Else, estimate a reasonable number of guesses s yielding Systems (7) with few solutions. For many parameters, the choice $s = 3$ turns out to be relevant.

3. Choose s *reference vectors* $(w_1, \dots, w_s) \in \mathbb{F}_2^n$ for which the values $\mathsf{rankJac}_{\mathrm{lin}}\, F(w_i)$ lie among the rare values in the rank distribution.

4. By brute-force search, guess an s–tuple $(v_1, \dots, v_s)$ such that $\mathsf{rankJac}_{\mathrm{lin}}\, G(v_i) = \mathsf{rankJac}_{\mathrm{lin}}\, F(w_i)$ for any $i \in \{1, \dots, s\}$. For each such guess, solve the system

$$\begin{cases} X \cdot \mathrm{Jac}_{\mathrm{lin}}\, G(v_i) - \mathrm{Jac}_{\mathrm{lin}}\, F(w_i) \cdot Y & = \quad 0 \\ \qquad\qquad Y \cdot v_i & = \quad w_i. \end{cases}, \qquad \forall i \in \{1, \dots, s\}.$$

5. If the above system has "too many solutions", make another guess. A threshold T for the dimension of the space of solutions should have been chosen. In our experiments, we set it to 10 in order to have at most 1024 solutions for the system. If the threshold is exceeded too frequently, the choice of s may be underestimated and relaunching the algorithm with a larger s would be relevant.

6. For each guess for which the space of solutions of the system is small enough, we perform brute-force search in this solution space toward a pair of matrices (X, Y) which are both non-singular. Such a pair provides a relevant candidate for (A_0^{-1}, B_0). For such a candidate, we use the calculations of Section 3.4.4 to deduce a 4–tuple (A_0, B_0, C_0, a). If it succeeds, we get our equivalence, if not, we keep on searching.

Remark 29. The reference vectors $(w_1, \dots, w_s)$ should be chosen linearly independent. Indeed, if some w_i is linearly linked with the other ones, then the contribution of w_i in the linear system (7) is useless, providing linear equations which are linked with the other ones.

Remark 30. It may happen that $\mathcal{R}_{\mathrm{dist}}(F)[r_0]$ is smaller than s or more generally that $\mathcal{R}(F)[r_0]$ does not contain s linearly independent elements. This situation is actually advantageous: suppose for instance that $s = 3$ and there are only 2 vectors w_1, w_2 in $\mathcal{R}(F)[r_0]$ and 2 vectors v_1, v_2 in $\mathcal{R}(G)[r_0]$. In this situation, we choose r_1 which is the index of the smallest entry larger than $\mathcal{R}_{\mathrm{dist}}(G)[r_0]$ in the rank distribution, choose $v_3 \in \mathcal{R}(G)[r_1]$ and perform a brute-force search for a $w_3 \in \mathcal{R}(F)[r_1]$ such that either $((v_1, w_1), (v_2, w_2), (v_3, w_3))$ or $((v_1, w_2), (v_2, w_1), (v_3, w_3))$ is a relevant guess. The number of guesses we should investigate is at most $2\mathcal{R}_{\mathrm{dist}}(F)[r_1]$ instead of $\mathcal{R}_{\mathrm{dist}}(F)[r_0]^3$.

Remark 31. As written, the algorithm might fail to return the solution. Indeed, if for a given guess $(v_1, \cdots, v_s)$, System (7) has a space of solutions whose dimension exceeds the threshold T, this guess is not further investigated. For this reason, the equivalence might be missed. A manner to address this issue would be to add some recursive call in this situation and brute forcing an $(s+1)$–th vector to guess in order to reduce the dimension of the solution space of (7).

3.4.6 Complexity

According to Lemma 21, the cost of the computation of the rank table is of $\mathrm{O}\left(\max(n, m)^\omega 2^n\right)$. Next, we have to evaluate the cost of the searching part of the algorithm. For any guess $v_1, \dots, v_s$, we have to solve a linear system of $m^2 + n^2$ unknowns and $\mathrm{O}\left(m^2 + n^2\right)$ equations; actually the choice of s is done so that this system has a small solution space and hence is close to be square.

The number of guesses we should perform is of $\mathrm{O}\left(R_1 \cdots R_s\right)$, where for any $i \in \{1, \dots, s\}$, $R_i := \mathcal{R}_{\mathrm{dist}}(F)[r_i]$ with $r_i := \mathsf{rankJac}_{\mathrm{lin}}\, F(w_i)$, the w_i's being the reference vectors defined in Section 3.4.5. In summary, for a random quadratic function F, denoting by $R = \max_i\{R_i\}$, the running time of the algorithm is of

$$\mathrm{O}\left(\max(n, m)^\omega 2^n + R^s (m^2 + n^2)^\omega\right),$$

Algorithm 1 Algorithm for EA-equivalence recovery.

- **Input:** A pair of Boolean functions $F, G : \mathbb{F}_2^n \longrightarrow \mathbb{F}_2^m$ and a threshold T (usually $T \leqslant 10$).
- **Output:** A 4-tuple (A_0, B_0, C_0, a) such that

$$\forall x \in \mathbb{F}_2^n,\ G(x) = A_0 \cdot F(B_0 x) + C_0 x + a$$

if it exists. Otherwise, returns "NOT EQUIVALENT" or "NO EQUIVALENCE FOUND".

Compute $\mathcal{R}(F)$ and $\mathcal{R}(G)$ and the corresponding rank distributions.
if $\mathcal{R}_{\text{dist}}(F) \neq \mathcal{R}_{\text{dist}}(G)$ **then**
    return "NOT EQUIVALENT"
end if
Let r_0 be the least occurring nonzero rank value.
Determine the number s of vectors to guess (in general $s = 3$ is enough).
Choose s linearly independent reference vectors $w_1, \ldots, w_s \in \mathcal{R}(F)[r_0]^s$.
for any s–tuple of linearly independent vectors $(v_1, \ldots, v_s) \in \mathcal{R}(G)[r_0]^s$ **do**
    Compute the solution space $\mathcal{S}$ of the system with variables $(X, Y) \in \mathbb{F}_2^{m\times m} \times \mathbb{F}_2^{n\times n}$

$$\left\{\begin{array}{rcl} X \cdot \mathrm{Jac}_{\mathrm{lin}}\, G(v_i) - \mathrm{Jac}_{\mathrm{lin}}\, F(w_i) \cdot Y & = & 0 \\ Y \cdot v_i & = & w_i. \end{array}\right. , \quad \forall i \in \{1, \ldots, s\}.$$

    if $\dim \mathcal{S} \leqslant T$ **then**
        for $(X, Y) \in \mathcal{S}$ **do**
            if both X and Y are non-singular **then**
                $A_0 := X^{-1}$ and $B_0 := Y$
                if Some 4–tuple (A_0, B_0, C_0, a) may be deduced using Section 3.4.4 **then**
                    return (A_0, B_0, C_0, a)
                end if
            end if
        end for
    end if
end for
return "NO EQUIVALENCE FOUND"

---

and we recall that in general $s = 3$ is a relevant choice. In practice, for random quadratic functions, the vectors providing the minimal entries of the rank distribution are very rare, permitting a very fast running of the algorithm (see Remark 30).

On the other hand, the situation where the algorithm is the least efficient is when the functions are APN. Indeed, as proved in Corollary 25, APN functions are precisely the ones whose nonzero ranks are all equal to $n - 1$. For such functions, the complexity of the algorithm is of

$$\mathrm{O}\left(n^{2\omega} 2^{sn}\right).$$

Here again, the choice $s = 3$ seems relevant. Note that, according to Remark 28, it may be possible that the choice $s = 2$ is sufficient and provides a number of guesses of 2^{2n} at the cost of some overhead for each step of the brute-force search. We have not been able to estimate the asymptotic behaviour of this overhead.

3.4.7 Examples of Running Times

The algorithm has been implemented using SageMath [S+19] and tested on a personal machine equipped with an Intel® Core(TM) i5-8250U CPU @ 1.60GHz. The source code is available on GitHub[2] Since the behaviour and the running time highly depends on the rank distribution, we list some examples of running time in Table 3. The examples take as input a pair (F, G) of quadratic functions where F is drawn at random and G is extended affine equivalent to F.

m	n	Rank distribution	Number of tries	Time (seconds)
6	6	[1, 0, 0, 2, 18, 43, 0]	21	0.68
6	6	[1, 0, 0, 1, 24, 38, 0]	386	5.36
6	6	[1, 0, 0, 0, 27, 36, 0]	4605	61.1
6	8	[1, 0, 0, 0, 9, 96, 150]	127	15.5
6	8	[1, 0, 1, 12, 98, 144]	24	13.8
8	6	[1, 0, 0, 0, 0, 63, 0]	11067	195.1
8	6	[1, 0, 0, 0, 3, 60, 0]	318	53.4
8	8	[1, 0, 0, 0, 0, 6, 93, 156, 0]	95	20.3
8	8	[1, 0, 0, 0, 1, 13, 104, 137, 0]	36	15.3

Table 3: This table lists some experiments launched on extended affine equivalent functions. All of them consisted in guessing $s = 3$ vectors (v_1, v_2, v_3). The fourth column gives the number of iterations, *i.e.* the number of guesses we made before finding a good triple (v_1, v_2, v_3).

4 Testing EA-Equivalence for Functions of any Degree

In this section, we describe how it is possible to test EA-equivalence for functions of any degree, while the algorithm described in the previous section was devoted to functions of degree 2. However, unlike above, we will focus on EA-testing rather than EA-recovery. To this end, we present several invariants in Section 4.1. With the exception of the one based on the properties of the ortho-derivative, they were all known before.

We then show how these invariants can used for partitioning the CCZ-class of a given function into EA-classes, and we detail two case-studies. First, Section 4.3 presents partitions of the CCZ-classes of all the 6-bit APN quadratic functions into EA-classes. Then, in order to compare the different invariants, we look at the various partitions that they define over the set of all known 8-bit known APN functions. Our results are in Section 4.5. As we will see, in the case of quadratic APN functions, the invariants based on the ortho-derivative are *by far* the finest grained.

4.1 Solving the EA-testing Problem

The general approach in solving the EA-testing problem consists in computing some quantities for each function that are invariant under EA-equivalence. The set of all these quantities is then used as a bucket label: if two distinct functions fall in different buckets, then they cannot be EA-equivalent. In the case where several functions are in the same bucket, we then need to solve the EA-recovery problem for each pair of functions in order to sort them in different EA-classes.

This general approach could be applied to other forms of equivalence, but we focus here on the cases of CCZ and extended affine equivalence. The invariants discussed in this paper are

[2] https://github.com/alaincouvreur/EA_equivalence_for_quadratic_functions

summarized in Table 4, along with their time complexities.[3] Note that other *ad hoc* invariants can be built, such as the one based on the presence of permutations in the EA-class that we apply to the CCZ-class of the Kim mapping in Section 4.3. Those listed in Table 4 are the most general ones.

Equivalence	Invariant	Condition	Complexity	Ref.
CCZ	Extended Walsh spectrum	–	$n2^{n+m}$	–
	Differential spectrum	–	2^{n+m}	–
	# Subspaces with dim $= n$ in $\mathcal{Z}_F$	–	?	[CP19]
	Γ-rank	$m = n$	$2^{\omega n}$	[BDKM09]
	Δ-rank	$m = n$	$2^{\omega n}$	[BDKM09]
EA	Algebraic degree	–	$m2^n$	–
	Thickness spectrum	–	?	[CP19]
	Σ^k-spectrum	k even	$2^{n(k-1)}$	[Kal20]
	# of subspaces in non-bent components	$\deg(F) = 2$	?	[BCC+20]
	Affine-equivalence class of π_F	$\deg(F) = 2$, APN, $m = n$	2^{n+m}	Sec. 4.1.2

Table 4: A summary of all the class invariants we are aware of.

4.1.1 Invariants from the Literature

CCZ-invariants. It is well-known that both the differential and the extended Walsh spectra are constant within a CCZ-class, and hence within an EA-class. It is also the case of the Γ-*rank* and the Δ-*rank* [BDKM09]. Let us define those quantities. First, for any set S of elements of $\mathbb{F}_2^n$, we let $\mathsf{Mat}(S)$ be the $|S| \times |S|$ binary matrix defined by

$$\mathsf{Mat}(S)[x, y] = 1 \;\Leftrightarrow\; x + y \in S \ .$$

Definition 32 (Γ- and Δ-rank)**.** *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a function. We call* Γ-rank *of F the rank of the matrix*

$$\mathsf{Mat}\left(\{(x, F(x)) : x \in \mathbb{F}_2^n\}\right)$$

and we call Δ-rank *the rank of the matrix*

$$\mathsf{Mat}\left(\{(a, b) : a, b \in \mathbb{F}_2^n, F(x + a) + F(x) = b \text{ has 2 solutions}\}\right) \ .$$

Those two invariants are only defined when $n = m$.

EA-invariants. While the previously mentioned quantities are invariant under CCZ-equivalence, the algebraic degree is constant within an EA-class, but not *a priori* within a CCZ-class. Another EA-invariant introduced in [CP19] is based on the linear subspaces which are contained in the *Walsh zeroes* of the function.

[3] The number of subspaces, the thickness spectrum, and the subspaces among non-bent components all rely on a vector space search which can be done using the algorithm from [BPT19]. However, as this algorithm is essentially a highly optimized tree search, its time complexity is hard to predict.

Definition 33 (Walsh Zeroes)**.** *[CP19, Def. 5] Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a function and let $\mathcal{W}_F$ be its Walsh transform, so that $\mathcal{W}_F(a, b) = \sum_{x \in \mathbb{F}_2^n} (-1)^{a \cdot x + b \cdot F(x)}$. We call* Walsh zeroes *of F the set*

$$\mathcal{Z}_F = \{(a, b) \in \mathbb{F}_2^n \times \mathbb{F}_2^m, \mathcal{W}_F(a, b) = 0\} \cup \{(0, 0)\} .$$

The *thickness spectrum* is then an EA-invariant defined in [CP19, Def. 9] which is derived from the structure of the Walsh zeroes.

Definition 34 (Thickness Spectrum)**.** *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a function, and let $\mathcal{Z}_F$ be its Walsh zeroes. Furthermore, let $\{V_i\}_{0 \leqslant i < \ell}$ be the set of all ℓ vector spaces of dimension n that are contained in $\mathcal{Z}_F$. The* thickness spectrum *of F is the set of positive integers $\{N_j\}_{0 \leqslant j \leqslant n}$ which are such that there are exactly N_j spaces V_i with a projection of dimension j on $\{(0, x), x \in \mathbb{F}_2^m\}$. This dimension is the* thickness *of the space.*

Recently, another multiset which is again an EA-invariant but not a CCZ-invariant was presented by Kaleyski [Kal20].

Definition 35 (Σ^k-spectrum)**.** *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a function, k be an even integer, and let $\Sigma_k^F(t)$ be defined for any $t \in \mathbb{F}_2^n$ as*

$$\Sigma_k^F(t) \;=\; \left\{ \sum_{i=0}^{k-1} F(x_i) : \{x_0, ..., x_{k-1}\} \subseteq \mathbb{F}_2^n, \quad \textit{and} \quad \sum_{i=0}^{k-1} x_i = t \right\} .$$

We then call Σ^k-spectrum *of F the sequence $\{(j, N_j)\}_{j \in J}$ which is such that exactly N_j distinct values appear j times in $\Sigma_k^F(0)$.*

As established in Proposition 1 of [Kal20], the Σ^k-spectrum is an EA-class invariant when $k > 2$ is even. On the other hand, it is easy to verify experimentally that it is not a CCZ-class invariant. The running time of the basic algorithm evaluating the Σ^k-spectrum is proportional to $2^{n(k-1)}$, meaning that the value of k is highly constrained by efficiency considerations.

If the function under consideration has some bent components, then the number of vector subspaces of a given dimension that are contained within the set of such components is also an EA-invariant. Similarly, the number of subspaces contained in the set of non-bent components is also constant within an EA-class, as mentioned in [BCC+20].

Functions computing all these invariants in the case where $m = n$ have been added to the public `sboxU` library.[4]

4.1.2 Invariants of Quadratic APN Functions Based on the Ortho-Derivative

A highly specific but very common case of EA-testing consists in determining whether two quadratic APN functions are EA-equivalent. The EA-recovery algorithm described in Section 3 can then used but we could hope for a faster algorithm which efficiently distinguishes most EA-classes without recovering the involved triple of affine functions (A, B, C). At this aim, we use a notion related to the derivatives of a quadratic function. This concept has already been used in several works, e.g. [CCZ98, Kyu07, Gor19, Gor20], but without a well-defined name.

Definition 36. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic function. We say that $\pi : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is an* ortho-derivative *for F if, for all x and a on $\mathbb{F}_2^n$,*

$$\pi(a) \cdot \big(F(x) + F(x + a) + F(0) + F(a)\big) = 0 .$$

[4] https://github.com/lpp-crypto/sboxU

Intuitively, the fact that F is quadratic implies that $\nabla_a : x \mapsto F(x) + F(x+a) + F(0) + F(a)$ is linear, and thus that its image set is a vector space with a well-defined orthogonal.

Since a quadratic function F is APN if and only if the sets $\{F(x)+F(x+a)+F(0)+F(a), x \in \mathbb{F}_2^n\}$ are hyperplanes for all nonzero $a \in \mathbb{F}_2^n$, we immediately deduce the following result.

Lemma 37. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic function. Then, F is APN if and only if it has unique ortho-derivative π such that $\pi(0) = 0$ and $\pi(x) \neq 0$ for all nonzero x.*

From now on, we will focus on quadratic APN functions and on the unique ortho-derivative defined as in the previous lemma. This function is strongly related to the Jacobian matrix introduced in Section 3 as explained in the following statement.

Proposition 38. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN function. For any $a \in \mathbb{F}_2^n \setminus \{0\}$, the vector $\pi_F(a)$ is the unique nonzero vector of the left kernel of $\mathrm{Jac}_{\mathrm{lin}}\, F(a)$.*

Proof. From Definition 36 together with Proposition 14, we have

$$\forall x \in \mathbb{F}_2^n, \quad \pi_F(a) \cdot (F(x) + F(x+a) + F(0) + F(a)) = \pi_F(a) \cdot (\mathrm{Jac}_{\mathrm{lin}}\, F(x) \cdot a) = 0$$

From Corollary 15, this identity becomes

$$\forall x \in \mathbb{F}_2^n, \quad (\pi_F(a) \cdot \mathrm{Jac}_{\mathrm{lin}}\, F(a)) \cdot x = 0$$

Thus, the vector $\pi_F(a) \cdot \mathrm{Jac}_{\mathrm{lin}}\, F(a)$ is orthogonal to any $x \in \mathbb{F}_2^n$ and hence is zero. □

Proposition 39. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN function and let π_F be its ortho-derivative. Furthermore, let A and B be affine permutations of $\mathbb{F}_2^n$ and $C : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an affine function. Finally, let A_0 and B_0 be the linear parts of A and B respectively. Then the ortho-derivative of $G : x \mapsto (A \circ F \circ B)(x) + C(x)$ is*

$$\pi_G = (A_0^T)^{-1} \circ \pi_F \circ B_0 \ .$$

Proof. Thanks to Proposition 38 we only have to prove that $(A_0^T)^{-1} \circ \pi_F \circ B_0(a)$ is in the right kernel of $\mathrm{Jac}_{\mathrm{lin}}\, G(a)$ for any $a \in \mathbb{F}_2^n$. Let $a \in \mathbb{F}_2^n \setminus \{0\}$, we have

$$\left(\left((A_0^T)^{-1} \circ \pi_F \circ B_0\right)(a)\right)^T \cdot \mathrm{Jac}_{\mathrm{lin}}\, G(a) = \pi_F(B_0 a)^T \cdot A_0^{-1} \cdot \mathrm{Jac}_{\mathrm{lin}}\, G(a).$$

From Proposition 17, $\mathrm{Jac}_{\mathrm{lin}}\, G(a) = A_0 \cdot \mathrm{Jac}_{\mathrm{lin}}\, F(B_0 a) \cdot B_0$ and hence

$$\begin{aligned}\left(\left((A_0^T)^{-1} \circ \pi_F \circ B_0\right)(a)\right)^T \cdot \mathrm{Jac}_{\mathrm{lin}}\, G(a) &= \pi_F(B_0 a)^T \cdot A_0^{-1} \cdot A_0 \cdot \mathrm{Jac}_{\mathrm{lin}}\, F(B_0 a) \cdot B_0 \\ &= \pi_F(B_0 a)^T \cdot \mathrm{Jac}_{\mathrm{lin}}\, F(B_0 a) \cdot B_0,\end{aligned}$$

and this last vector is zero by Definition 36 together with Proposition 14. □

Remark 40. A first immediate use of Proposition 39 would consist in solving the EA-recovery problem, i.e. in finding (A, B, C) such that $G = A \circ F \circ B + C$, by using an algorithm solving the *affine-equivalence*-recovery problem between the ortho-derivatives π_F and π_G. Several *affine-equivalence*-recovery algorithms exist in the literature, namely in [BDBP03] and in [Din18]. The former only works for permutations, which means that we can use it efficiently to test EA-equivalence when n is odd, since π_F and π_G are bijective in this case [CCZ98]. However, the ortho-derivative is not a bijection when n is even, meaning that we cannot use it. While the algorithm of [Din18] can efficiently handle non-bijective functions, and requires that their algebraic degree be *at least* $n-2$, we have found in practice that for $n \in \{6, 8, 10\}$, it in fact requires that the degree be $n-1$. Indeed, the algorithm fails for functions of degree $n-2$ (be they ortho-derivatives or not). As we have experimentally observed that ortho-derivatives are always of degree $n-2$ (see also [Gor20]), it is probably the reason why, in practice, this algorithm does not work either. Hence, to the best of our knowledge, the algorithm we presented in Section 3.4 is the only one solving the EA-recovery problem efficiently, even in the very specific case of quadratic APN functions.

Despite this limitation, Proposition 39 still gives us a very powerful tool to solve the EA-testing problem. Indeed, it implies that if F and G are EA-equivalent quadratic APN functions, then their ortho-derivatives have to be affine-equivalent. If π_F and π_G are not affine-equivalent, then F and G cannot be EA-equivalent (and thus CCZ-equivalent since both notions coincide when F and G are quadratic [Yos11]). In practice, as discussed in Section 4.5, we have found that the differential and extended Walsh spectra of the ortho-derivatives vary significantly, and in fact provide an EA-class invariant that is both the finest grained (by far), and can be computed very efficiently.

Note that the algebraic degree of the ortho-derivative cannot be used as an EA-class invariant. Indeed we have observed it be always equal to $n-2$, as conjectured by Gorodilova [Gor20].

4.2 Partitioning CCZ-classes into EA-classes

A very common use-case of EA-testing is when we want to obtain information about the CCZ-class of a function, especially when we want to partition a CCZ-class into EA-classes. Indeed, the technique presented in [CP19] enables us to loop through representatives of all the EA-classes in a CCZ-class. This method is derived from the following property related to the Walsh zeroes of the functions.

Proposition 41 ([CP19])**.** *A linear permutation $\mathcal{A}$ of $\mathbb{F}_2^{2n}$ is admissible for a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ if and only if $\mathcal{A}^T(\mathcal{V}) \subseteq \mathcal{Z}_F$ where $\mathcal{V} = \{(x,0), x \in \mathbb{F}_2^n\}$.*

As a consequence, it is possible to loop through representatives of all the EA-classes contained in the CCZ-class of a function F by identifying all the vector spaces of dimension n contained in $\mathcal{Z}_F$, deducing the admissible mapping corresponding to each of them, and then applying it to the graph of F. This theoretical approach can be implemented efficiently using the vector space search algorithm presented in [BPT19]. However, while it allows a full exploration of the EA-classes contained in the CCZ-class, it may return several representatives for a given EA-class. In other words, several functions obtained with this method may lie in the same EA-class. This situation can then be detected by using some of the previously mentioned EA-invariants.

We will now use our EA-testing algorithms for studying the EA-classes included in the CCZ-classes of all 6-bit APN quadratic functions, with a particular focus on the EA-classes which contain permutations.

We will then need the following result established in [CP19].

Proposition 42. *A function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a permutation if and only if $\mathcal{V} \subset \mathcal{Z}_F$ and $\mathcal{V}^\perp \subset \mathcal{Z}_F$, where*

$$\mathcal{V} = \{(x,0), x \in \mathbb{F}_2^n\}, \quad \textit{and} \quad \mathcal{V}^\perp = \{(0,x), x \in \mathbb{F}_2^n\} .$$

4.3 Kim Mapping and Dillon et al.'s Permutation

Let $n = m = 6$. The Kim mapping is a quadratic APN function κ defined over $\mathbb{F}_{2^6}$ by $\kappa(x) = x^3 + x^{10} + wx^{24}$, where w is a root of the primitive polynomial used to define $\mathbb{F}_{2^6}$. It is well-known for being CCZ-equivalent to a permutation since it is the function which served as a basis for the result of Dillon et al. [BDMW10].

We ran the bases extraction algorithm of [BPT19] on $\mathcal{Z}_\kappa$ and found that it contains a total of 222 distinct vector spaces of dimension 6. We then deduce that the CCZ-class of the Kim mapping contains at most 222 EA-classes. We generated representatives of these 222 possibly distinct EA-classes and we computed their respective thickness spectra. We have found 8 different thickness spectra, showing that at least 8 of these EA-classes are distinct.

Let us now focus on the EA-classes within the CCZ-class of the Kim mapping which contain permutations. By calculating the dimension of the projection on $\mathcal{V}^\perp$ of each of these 222 spaces, we obtain the thickness spectrum of κ:

$$N_0 = 1,\ N_1 = 63,\ N_2 = 126,\ N_3 = 32\ .$$

To enumerate the EA-classes that are containing permutations, it is necessary and sufficient to find pairs (U, V) of vector spaces such that $U \cup V$ spans the full space $(\mathbb{F}_2^n)^2$ (Proposition 42). Indeed, we then simply need to construct a linear permutation L of $(\mathbb{F}_2^n)^2$ such that $L(U) = \mathcal{V}$ and $L(V) = \mathcal{V}^\perp$, and then to apply L^T to the graph $\Gamma_F = \{(x, F(x)), x \in \mathbb{F}_2^n\}$ to obtain the graph $\Gamma_G = L^T(\Gamma_F)$ of a permutation G.

As the dimension of $\mathcal{V}^\perp$ here is 6, the only spaces that could be used to construct such pairs have a thickness of 3. In fact, there exist two families of 16 vector spaces of dimension 6 which we denote $\{V_i\}_{i<16}$ and $\{U_i\}_{i<16}$, and which are such that $\langle V_i \cup U_j \rangle = (\mathbb{F}_2^n)^2$ for any i, j. The following proposition will allow us to leverage these permutations to identify two distinct EA-classes among the corresponding permutations.

Proposition 43. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and $F' : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be two CCZ-equivalent functions, and suppose that there exist two families $\{L_i\}_{0 \leqslant i < \ell}$ and $\{L'_i\}_{0 \leqslant i < \ell'}$ of linear functions such that:*

- $\ell > 0$ *or* $\ell' > 0$,
- $F + L_i$ *is a permutation for all* $i < \ell$, *and*
- $F' + L'_i$ *is a permutation for all* $i < \ell'$.

Suppose moreover that the families $\{L_i\}_{0 \leqslant i < \ell}$ and $\{L'_i\}_{0 \leqslant i < \ell}$ are maximal *for this property,* i.e. *any family of linear functions $\{M_j\}_j$ (resp. $\{M'_j\}_j$) satisfying the above properties is contained in $\{L_i\}_{0 \leqslant i < \ell}$ (resp. $\{L'_i\}_{0 \leqslant i < \ell}$). Then F and F' are EA-equivalent if and only if $\ell = \ell'$ and if there exists a permutation σ of $\{0, ..., \ell - 1\}$ such that $F + L_i$ is affine-equivalent to $F' + L'_{\sigma(i)}$ for all $i < \ell$.*

Proof. If F and F' are EA-equivalent then it is clear that $\ell = \ell'$ and that such a permutation σ exists. Let us then focus on the opposite, and suppose that $\ell = \ell'$ and that there exists a permutation σ such that $F + L_i$ is affine-equivalent to $F' + L'_{\sigma(i)}$. Then in particular there exist i and j such that $F + L_i$ is affine-equivalent to $F' + L'_j$. As a consequence, there also exist affine permutations A and B such that $F + L_i = B \circ (F' + L'_j) \circ A$, which is equivalent to

$$F = B \circ F' \circ A + \underbrace{B \circ L'_j \circ A + B(0) + L_i}_{C'}\ .$$

We then deduce that F and F' are EA-equivalent. $\square$

We generated the 256 permutations obtained by mapping (V_i, U_j) to $(\mathcal{V}, \mathcal{V}^\perp)$ for all $(i, j) \in \{0, ..., 15\}^2$ and their inverses obtained by mapping (U_i, V_j) to $(\mathcal{V}, \mathcal{V}^\perp)$. Using the algorithm of Biryukov et al. [BDBP03], we found that these 512 permutations fall into only four distinct affine equivalence classes. We denote these four affine equivalence classes by $\mathcal{A}_k$ for $k \in \{0, 1, 2, 3\}$.

Let us first exhibit the permutations living in these affine-equivalence classes. Recall that the so-called *generalized open butterfly* as introduced in [CDP17] is a family of permutations which contains in particular some APN permutations for $n = 6$. It was obtained by generalizing the structure first identified in [PUB16]. These permutations are parameterised by two finite-field elements α and β of $\mathbb{F}_{2^3}$. They are the involutions defined as $\mathsf{H}_{\alpha,\beta} : (\mathbb{F}_{2^3})^2 \to (\mathbb{F}_{2^3})^2$, where

$$\mathsf{H}_{\alpha,\beta}(x, y) = \left(T_y^{-1}(x),\ T_{T_y^{-1}(x)}(y)\right) \text{ and } T_y(x) = (x + \alpha y)^3 + \beta y^3 .$$

We experimentally found that the four affine equivalence classes $\mathcal{A}_k$, $k \in \{0, \dots, 3\}$, contain the following representatives, where $\alpha \neq 0$, $\mathrm{Tr}(\alpha) = 0$:

- $\mathcal{A}_0$ contains $\mathsf{H}_{\alpha,1}$,
- $\mathcal{A}_1$ contains $\mathsf{H}_{\alpha,\beta}$ with $\beta = \alpha^3 + 1/\alpha$,
- $\mathcal{A}_2$ contains the permutations $P = \mathsf{H}_{\alpha,1} + L$ such that L is linear and $P \notin \mathcal{A}_0$,
- $\mathcal{A}_3$ contains the permutations $P' = \mathsf{H}_{\alpha,\beta} + L$ such that L is linear and $P' \notin \mathcal{A}_1$.

Proposition 43 imposes the existence of at least two EA-classes containing permutations within the CCZ-class of the Kim mapping: one that contains $\mathcal{A}_0$ and $\mathcal{A}_2$, and another one that contains $\mathcal{A}_1$ and $\mathcal{A}_3$. Indeed, if $\mathcal{A}_0$ and $\mathcal{A}_1$ were EA-equivalent, then any permutation of the form $(\mathsf{H}_{\alpha,1} + L)$ in $\mathcal{A}_0 \cup \mathcal{A}_2$ would be affine-equivalent to some $(\mathsf{H}_{\alpha,\beta} + L')$, while all such functions belong to another affine-equivalence class, included in $\mathcal{A}_1 \cup \mathcal{A}_3$. Since our vector-space-based approach enumerated all EA-classes (possibly multiple times), and since all the representatives of EA-classes containing permutations ended up in one of these two EA-classes, we can conclude that there exists exactly two EA-classes of permutations in this CCZ-class. Furthermore, we also found out that if $P \in \mathcal{A}_0$ then $P^{-1} \in \mathcal{A}_0$. The same holds for $\mathcal{A}_1$. On the other hand, if $P \in \mathcal{A}_2$ then $P^{-1} \in \mathcal{A}_3$, and vice-versa.

Lemma 44. *Up to extended-affine equivalence, all known APN permutations in even dimension are generalized open butterflies in the sense of [CDP17]. All these known APN permutations in even dimension belong to exactly two EA-classes.*

We also remark that the thickness spectrum of the two EA-classes containing permutations is the same, namely

$$N_0 = 1,\ N_1 = 7,\ N_2 = 14,\ N_3 = 58,\ N_4 = 42,\ N_5 = 84,\ N_6 = 16 .$$

Thus, while having different thickness spectra implies being in distinct EA-classes, the converse is not true. They also share the same Σ^4-spectrum, meaning that the same observation applies to this invariant.

Picture Representation. All these results are summarized in Figure 1, which contains a graphical representation of the CCZ-class of the Kim mapping. It is partitioned into 8 parts, each corresponding to a different thickness spectrum. We also specified the algebraic degree d in each of these parts. The Kim mapping itself is in the only quadratic part. Further, using the main result of [Yos11], we can claim that this part corresponds to a unique EA-class.

At this stage, we cannot know how many EA-classes are in each of the other parts, except for the one containing permutations. As discussed above, it contains two distinct EA-classes: one containing $\mathsf{H}_{\alpha,\beta}$, and one containing $\mathsf{H}_{\alpha,1}$. The border between these two EA-classes is represented by a dashed line, while their affine-equivalence classes are represented with circles.

We used blue arrows to represent the mappings called t-twists (see [CP19]) that send the Kim mapping to each part of the CCZ-class. The value of t is given, and we use different lines for different t as well. For example, since the open butterflies are involutions, a 6-twist (which is the same as an inversion) maps these functions to themselves. Similarly, each EA-class containing permutations is obtained from a function EA-equivalent to the Kim mapping via a 3-twist.

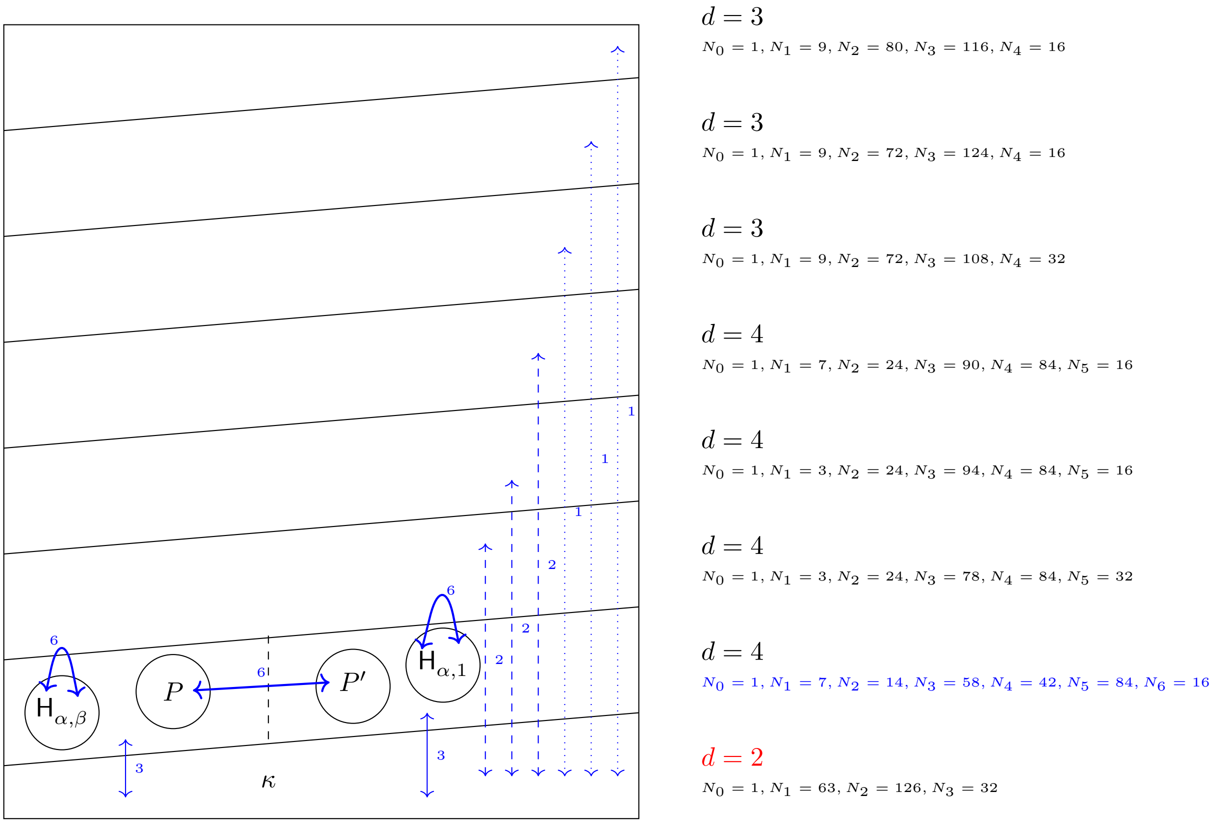

Figure 1: The overall structure of the CCZ-class of the Kim mapping κ. Each of the 7 upper parts may contain several EA-classes. The circles correspond to the four affine-equivalence classes of permutations, and arrows correspond to t-twists.

4.4 6-Bit Quadratic APN Functions.

We looked at the Banff list of the 13 different 6-bit quadratic APN functions (including the Kim mapping) which can be found for instance in [BN15] and which is recalled in Table 5. In Table 6, we list many properties of the Banff functions, namely their Δ-rank, Γ-rank, thickness spectra, as well as upper and lower bounds on the number of EA-classes in their CCZ-classes.

The upper bound is simply the number of vector spaces of dimension 6 in their Walsh zeroes. The lower bound is obtained for each function F by iterating through all the vector spaces V_i of dimension 6 in its Walsh zeroes, generating a linear permutation $\mathcal{L}$ such that $\mathcal{L}(\mathcal{V}) = V_i$, and then computing the thickness spectrum of the function G such that $\Gamma_G = \mathcal{L}(\Gamma_F)$. Since the thickness spectrum is constant in an EA-class, two functions with different thickness spectra must be in distinct EA-classes. Thus, the lower bound is the number of distinct thickness spectra obtained in this fashion. For the Kim mapping (number 5 in the list), we increase this number by 1 because we have established above that two distinct EA-classes share the same thickness spectrum.

There is a total of 7 distinct thickness spectra among these functions.

i	Univariate representation
1	x^3
2	$x^3 + \alpha^{11}x^6 + \alpha x^9$
3	$\alpha x^5 + x^9 + \alpha^4 x^{17} + \alpha x^{18} + \alpha^4 x^{20} + \alpha x^{24} + \alpha^4 x^{34} + \alpha x^{40}$
4	$\alpha^7 x^3 + x^5 + \alpha^3 x^9 + \alpha^4 x^{10} + x^{17} + \alpha^6 x^{18}$
5	$x^3 + x^{10} + \alpha x^{24}$
6	$x^3 + \alpha^{17}x^{17} + \alpha^{17}x^{18} + \alpha^{17}x^{20} + \alpha^{17}x^{24}$
7	$x^3 + \alpha^{11}x^5 + \alpha^{13}x^9 + x^{17} + \alpha^{11}x^{33} + x^{48}$
8	$\alpha^{25}x^5 + x^9 + \alpha^{38}x^{12} + \alpha^{25}x^{18} + \alpha^{25}x^{36}$
9	$\alpha^{40}x^5 + \alpha^{10}x^6 + \alpha^{62}x^{20} + \alpha^{35}x^{33} + \alpha^{15}x^{34} + \alpha^{29}x^{48}$
10	$\alpha^{34}x^6 + \alpha^{52}x^9 + \alpha^{48}x^{12} + \alpha^6 x^{20} + \alpha^9 x^{33} + \alpha^{23}x^{34} + \alpha^{25}x^{40}$
11	$x^9 + \alpha^4 x^{10} + \alpha^9 x^{12} + \alpha^4 x^{18} + \alpha^9 x^{20} + \alpha^9 x^{40}$
12	$\alpha^{52}x^3 + \alpha^{47}x^5 + \alpha x^6 + \alpha^9 x^9 + \alpha^{44}x^{12} + \alpha^{47}x^{33} + \alpha^{10}x^{34} + \alpha^{33}x^{40}$
13	$\alpha x^6 + x^9 + \alpha x^{10} + \alpha^4 x^{17} + \alpha x^{24} + \alpha x^{33}$

Table 5: The Banff list of quadratic APN functions operating on 6 bits.

i	Thickness Spectrum	Linearity	rank		# EA		Diff. Spec. of π_F
			Γ	Δ	min	max	
1	$N_0 = 1,\ N_1 = 63,\ N_2 = 126$	16	1102	94	3	190	$\{0:2205, 2:1764, 8:63\}$
2	$N_0 = 1,\ N_1 = 63,\ N_2 = 126$	16	1146	94	3	190	$\{0:2583, 2:1008, 4:378, 8:63\}$
3	$N_0 = 1,\ N_1 = 63,\ N_2 = 30$	16	1158	96	4	94	$\{0:2454, 2:1176, 4:370, 6:30, 10:2\}$
4	$N_0 = 1,\ N_1 = 63,\ N_2 = 42$	16	1166	94	5	106	$\{0:2338, 2:1428, 4:210, 6:56\}$
5	$N_0 = 1,\ N_1 = 63,\ N_2 = 126,\ N_3 = 32$	16	1166	96	**8+1**	222	$\{0:2373, 2:1428, 4:168, 8:63\}$
6	$N_0 = 1,\ N_1 = 63,\ N_2 = 54$	16	1168	96	9	118	$\{0:2442, 2:1229, 4:303, 6:51, 8:7\}$
7	$N_0 = 1,\ N_1 = 63,\ N_2 = 30$	**32**	1170	96	6	94	$\{0:2401, 2:1371, 4:195, 6:50, 14:15\}$
8	$N_0 = 1,\ N_1 = 63,\ N_2 = 42$	16	1170	96	8	106	$\{0:2426, 2:1255, 4:297, 6:49, 8:5\}$
9	$N_0 = 1,\ N_1 = 63,\ N_2 = 54$	16	1170	96	9	118	$\{0:2439, 2:1235, 4:297, 6:57, 8:4\}$
10	$N_0 = 1,\ N_1 = 63,\ N_2 = 54$	16	1170	96	9	118	$\{0:2422, 2:1271, 4:279, 6:53, 8:7\}$
11	$N_0 = 1,\ N_1 = 63,\ N_2 = 42,\ N_3 = 8$	16	1172	96	20	114	$\{0:2385, 2:1339, 4:258, 6:45, 8:2, 12:3\}$
12	$N_0 = 1,\ N_1 = 63,\ N_2 = 54,\ N_3 = 8$	16	1172	96	20	126	$\{0:2404, 2:1307, 4:261, 6:53, 8:7\}$
13	$N_0 = 1,\ N_1 = 63,\ N_2 = 42$	16	1174	96	9	106	$\{0:2414, 2:1271, 4:303, 6:37, 8:7\}$

Table 6: Several CCZ-class invariants for the functions in the Banff list and bounds on the number of EA classes in their CCZ-classes.

Combining all the invariants listed in Table 4 that are not based on the ortho-derivative, we still could not see that all these functions fit into different EA-classes as they are identical for Functions 9 and 10. However, as we can see, the differential spectrum of the ortho-derivative is sufficient on its own to show that they are indeed in different EA-classes. We could also look at the extended Walsh spectra of the ortho-derivative for a finer grained view, but it is not necessary here.

Interestingly, all ortho-derivatives have the trivial thickness spectrum (i.e. $\{N_0 = 1\}$), except for the cube mapping and for the Kim mapping. We have that

$$\text{thickness spectrum of } \ \pi_{x^3} = N_0 = 1, N_3 = 9, N_6 = 54$$
$$\text{thickness spectrum of } \ \pi_{\kappa} = N_0 = 1, N_6 = 5 \ ,$$

implying that both are EA-equivalent to permutations.

4.5 8-bit Quadratic APN functions

As Dillon et al. derived their APN permutation of 6 variables from a quadratic APN function, several teams have tried to reproduce this general approach by finding ways to generate large numbers of quadratic APN functions of an even number of variables, and then checking if they are in fact CCZ-equivalent to a permutation. While the answer to the latter question has always been no, we can now work with more than $20,000$ distinct 8-bit quadratic APN functions, the first $8,000$ having been obtained using the QAM [YWL14], and the next $12,000$ through an optimized guess-and-determine approach focusing on functions with internal symmetries [BL20a, BL20b].

Combining both lists gave us $21,102$ distinct quadratic APN functions. It turns out that all of these functions can be put into distinct buckets in a few minutes using the extended Walsh spectrum and differential spectrum of their ortho-derivative as a discriminant. The fact that these functions are thus in distinct CCZ-equivalence class[5] is not a new result,[6] but the speed of our method is noteworthy. It has a time complexity which is linear in the number of functions considered, a low memory complexity, and can handle tens of thousands of 8-bit functions in a few minutes on a desktop computer. We thus claim that our ortho-derivative-based approach is the best solution to the EA-testing problem in the case of quadratic APN functions. While this setting may be narrow, it is arguably one of the most interesting ones.

We can still use the other invariants to learn more about these functions. First, there are only 6 distinct extended Walsh spectra in the whole list:

$$\begin{gathered}\{0:16320,\ 16:43520,\ 32:5440\}\\ \{0:15600,\ 16:44544,\ 32:5120,\ 64:16\}\\ \{0:14880,\ 16:45568,\ 32:4800,\ 64:32\}\\ \{0:14160,\ 16:46592,\ 32:4480,\ 64:48\}\\ \{0:13440,\ 16:47616,\ 32:4160,\ 64:64\}\\ \{0:12540,\ 16:48640,\ 32:4096,\ 128:4\}\ ,\end{gathered}$$

meaning that there are many functions with identical extended Walsh spectra but distinct thickness spectra. On the other hand, some functions have identical thickness spectra but different Walsh ones (see the bottom of Table 7 for an example). There are $255 = 2^8 - 1$ different thickness spectra, a number which looks interesting in itself. Indeed, recall that there are $7 = 2^3 - 1$ different thickness spectra among all 6-bit quadratic APN functions.

We can also fit all these functions into 486 different buckets with distinct extended Walsh spectrum/thickness spectrum pairs. However, the functions are not uniformly spread among said buckets, in fact only 10 of these classes account for about a third of all functions (see the first rows of Table 7). We remark that, for all these large classes, the number of spaces of thickness 2 is always a multiple of 6, and that all such numbers comprised between 108 and 162 form the 10 most common occurrences. However, it is not necessary for N_2 to be a multiple of 6 as evidenced for example by the function with thickness spectrum such that $N_2 = 104 \equiv 2 \mod 6$.

At the opposite end, 143 functions live in classes that contain only one function. For instance, the function with the highest number of vector spaces of dimension n in its Walsh zeroes (669 in total) is alone in its bucket. Only two functions have spaces of thickness 4 in their Walsh zeroes.

All functions have $N_1 = 255$, a quantity which was explained to be related to the derivatives of quadratic functions in [CP19]. Interestingly, there are 22 functions for which there is nothing else in the thickness spectrum. The most prominent function in this set is the cube mapping $x \mapsto x^3$.

[5] Recall that CCZ-equivalence and EA-equivalence coincide in the case of quadratic functions [Yos11].

[6] In fact, the authors of [BL20a] used our method based on the ortho-derivative—and indeed our implementation—to solve this problem.

Cardinality	Extended Walsh spectrum	Thickness Spectrum
617	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 162$
681	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 156$
617	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 150$
606	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 144$
640	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 138$
681	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 132$
635	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 126$
664	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 120$
639	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 114$
616	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 108$
1	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 104$
22	$\{0 : 15600, 16 : 46520, 32 : 5440\}$	$N_0 = 1, N_1 = 255$
1	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 294, N_3 = 56, N_4 = 64$
1	$\{0 : 16320, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 210, N_3 = 56, N_4 = 64$
78	$\{0 : 15600, 16 : 43520, 32 : 5440\}$	$N_0 = 1, N_1 = 255, N_2 = 194$
9	$\{0 : 15600, 16 : 44544, 32 : 5120, 64 : 16\}$	$N_0 = 1, N_1 = 255, N_2 = 194$

Table 7: The properties of some interesting classes of 8-bit quadratic APN functions.

There is a wide variety of spectra of the shape $N_0 = 1, N_1 = 255, N_2 = \ell$ as ℓ varies from 12 to 264. We give the number of functions with each such thickness spectrum in Figure 2 (the Walsh spectra are not taken into account in this figure). As we can see, most functions verify $N_2 \equiv 0 \mod 6$, and the distribution of such functions seems to follow a Gaussian distribution with mean 132.06. There are fewer functions satisfying $N_2 \not\equiv 0 \mod 6$, and those seem to follow their own Gaussian distribution with a different mean of 166.50.

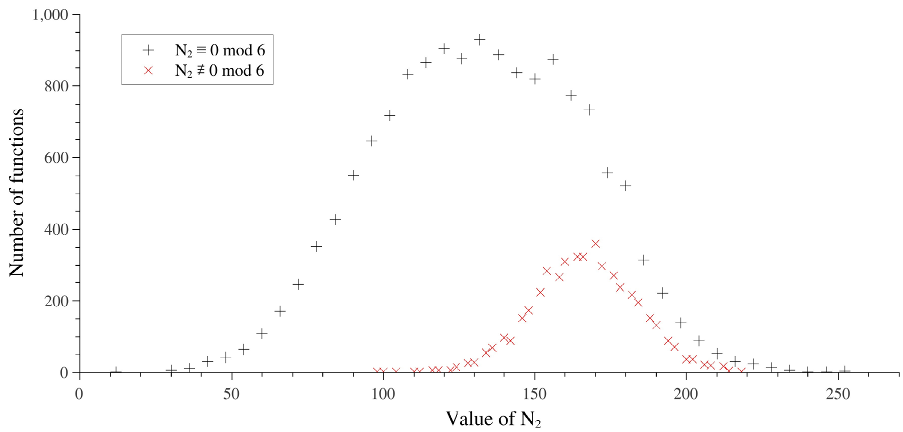

Figure 2: The number of known 8-bit quadratic APN functions such that the spaces in their Walsh zeroes have a maximum thickness of 2. Different symbols are used depending on whether $N_2 \equiv 0 \mod 6$ or not.

Finally, we remark that the ortho-derivatives of all the more than 20,000 functions we investigated have a trivial thickness spectrum, i.e. $\{N_0 = 1\}$.

5 Conclusion

We can efficiently solve both EA-recovery and EA-testing in a new set of cases that includes those that are of the most importance to researchers working on the big APN problem. In particular, our use of the ortho-derivative of quadratic APN functions for EA-testing has already enabled new results in this area [BL20a].

However, a general solution to both problems that could be applied in all cases, without conditions on the algebraic degree of the functions or on the shape of the affine mappings involved, remains to be found.

References

[BCC+20] Lilya Budaghyan, Marco Calderini, Claude Carlet, Robert S. Coulter, and Irene Villa. Constructing APN functions through isotopic shifts. *IEEE Trans. Inf. Theory*, 66(8):5299–5309, 2020.

[BDBP03] Alex Biryukov, Christophe De Cannière, An Braeken, and Bart Preneel. A toolbox for cryptanalysis: Linear and affine equivalence algorithms. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 33–50. Springer, Heidelberg, May 2003.

[BDKM09] K. A. Browning, J.F. Dillon, R.E. Kibler, and M. T. McQuistan. APN Polynomials and Related Codes. *J. of Combinatorics, Information and System Sciences*, 34(1-4):135–159, 2009.

[BDMW10] K. A. Browning, J.F. Dillon, M. T. McQuistan, and A. J. Wolfe. An APN permutation in dimension six. In *Post-proceedings of the 9-th International Conference on Finite Fields and Their Applications*, volume 518, pages 33–42. American Mathematical Society, 2010.

[BK12] Lilya Budaghyan and Oleksandr Kazymyrov. Verification of restricted EA-equivalence for vectorial boolean functions. In Ferruh Özbudak and Francisco Rodríguez-Henríquez, editors, *Arithmetic of Finite Fields - WAIFI 2012*, pages 108–118, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[BL20a] Christof Beierle and Gregor Leander. New instances of quadratic APN functions. `arXiv:2009.07204 [cs.IT]`, 2020. https://arxiv.org/abs/2009.07204.

[BL20b] Christof Beierle and Gregor Leander. New Instances of Quadratic APN Functions in Dimension Eight, September 2020.

[BN15] Céline Blondeau and Kaisa Nyberg. Perfect nonlinear functions and cryptography. *Finite Fields and Their Applications*, 32:120–147, 2015. Special Issue : Second Decade of FFA.

[BPT19] Xavier Bonnetain, Léo Perrin, and Shizhu Tian. Anomalies and vector space search: Tools for S-box analysis. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 196–223. Springer, Heidelberg, December 2019.

[CCZ98] Claude Carlet, Pascale Charpin, and Victor Zinoviev. Codes, bent functions and permutations suitable for DES-like cryptosystems. *Designs, Codes and Cryptography*, 15(2):125–156, 1998.

[CDP17] Anne Canteaut, Sébastien Duval, and Léo Perrin. A generalisation of Dillon's APN permutation with the best known differential and nonlinear properties for all fields of size 2^{4k+2}. *IEEE Transactions on Information Theory*, 63(11):7575–7591, Nov 2017.

[CP19] Anne Canteaut and Léo Perrin. On CCZ-equivalence, extended-affine equivalence, and function twisting. *Finite Fields and Their Applications*, 56:209–246, 2019.

[Din18] Itai Dinur. An improved affine equivalence algorithm for random permutations. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 413–442. Springer, Heidelberg, April / May 2018.

[Gor19] Anastasiya Gorodilova. On the differential equivalence of APN functions. *Cryptogr. Commun.*, 11(4):793–813, 2019.

[Gor20] Anastasiya Gorodilova. A note on the properties of associated boolean functions of quadratic APN functions. `arXiv:2005.10788 [cs.DM]`, 2020. https://arxiv.org/abs/2005.10788.

[Kal20] Nikolay Kaleyski. Deciding EA-equivalence via invariants. SEquences and Their Applications 2020—SETA'20. Available online: https://seta-2020.org/assets/files/program/papers/paper-44.pdf., 2020.

[Kyu07] Gohar M. M. Kyureghyan. Crooked maps in $\mathbb{F}_{2^n}$. *Finite Fields and Their Applications*, 13(3):713–726, 2007.

[Nyb91] Kaisa Nyberg. Perfect nonlinear S-boxes. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 378–386. Springer, Heidelberg, April 1991.

[Nyb94] Kaisa Nyberg. Differentially uniform mappings for cryptography. In Tor Helleseth, editor, *EUROCRYPT'93*, volume 765 of *LNCS*, pages 55–64. Springer, Heidelberg, May 1994.

[PUB16] Léo Perrin, Aleksei Udovenko, and Alex Biryukov. Cryptanalysis of a theorem: Decomposing the only known solution to the big APN problem. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 93–122. Springer, Heidelberg, August 2016.

[S+19] W. A. Stein et al. *SageMath, the Sage Mathematics Software System (Version 8.6)*, 2019. http://www.sagemath.org.

[Yos11] Satoshi Yoshiara. Equivalences of quadratic APN functions. *Journal of Algebraic Combinatorics*, 35:461–475, September 2011.

[YWL14] Yuyin Yu, Mingsheng Wang, and Yongqiang Li. A matrix approach for constructing quadratic APN functions. *Designs, Codes and Cryptography*, 73(2):587–600, Nov 2014.