Exploring the Set of APN Functions

Jules Baudrin[1], Pierre Galissant[2] and Léo Perrin[2]

[1] Université Paris-Saclay, UVSQ, CNRS, Laboratoire de Mathématiques de Versailles, Versailles, France
name.surname@uvsq.fr
[2] Inria, Paris, France
name.surname@inria.fr

Abstract. An Almost Perfect Nonlinear (APN) function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ provides optimal protection against differential attacks, meaning that the equation $F(x+a) = F(x)+b$ has at most 2 solutions x when $a \neq 0$. Setting aside some sporadic examples, such functions are hard to find in practice, and whether there exists APN bijections of $\mathbb{F}_2^n$ when $n > 6$ is even (the *Big APN Problem*) is still unknown.

In this paper, we provide the tools needed to explore the complete set of *known* APN functions. Indeed, for a given APN function F, we in fact know many: all functions *CCZ-equivalent* to F are also APN, and so are its *switching neighbours*. These notions have been in the literature for many years, but several unknowns remain that complicate the exploration of the corresponding functions in practice: a given function has too many switching neighbours, and the algorithms that are known to iterate through a CCZ-equivalence class return multiple representatives per extended-affine (EA) equivalence class.

We solve both problems. First, we show how to partition precisely the CCZ-equivalence class of a quadratic APN function in practice, and thus obtain exactly one representative per EA-equivalence class. To achieve this, we carefully investigate the set of the automorphisms of the graph of a quadratic APN function, and show how to generate them very efficiently using the ortho-derivative. Second, we provide the first detailed investigation of the set of the switching neighbours of a function, show in particular how an overwhelming majority of them are trivial, and explain how to get rid of those. Incidentally, we build a new extended-affine equivalence class invariant using the resulting set of non-trivial switching neighbours.

For both theoretical contributions, we pushed highly-efficient and easy to re-use implementations of the corresponding algorithms to the `sboxU` library. It allowed us to generate databases containing all known APN functions for $n = 6$ and $n = 7$. For $n = 8$, our algorithms are efficient-enough that we can investigate the properties of the millions of quadratic APN function recently exhibited. In the process, we generate a few thousands of new CCZ-equivalence classes.

In the light of our work, the successes of the exploration strategies for $n = 6$ (*i.e.* finding an APN function inequivalent to any monomial and quadratic mapping, and finding an APN bijection in even dimension) seems to be an artifact due to the small dimension that we show not to work for dimension 8 and which we do not expect to work either when the dimension is greater.

Keywords: APN functions · switching neighbours · CCZ equivalence · EA equivalence · automorphisms

1 Introduction

Context. Most of the functions appearing in the context of symmetric cryptography map n-bit words to m-bit words (where m can possibly be equal to n). In this specific context, the cryptanalyst is particularly interested in the cryptographic properties of functions of the form $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ known as vectorial Boolean functions. More precisely, whenever new attack paths against existing primitives are discovered, (un)desirable properties are most of the time identified and influence the design strategies of future constructions.

The main example is indubitably the case of differential attacks [BS91, BS93] which takes advantage of some pair of differences $(\Delta^{\text{in}}, \Delta^{\text{out}}) \in \mathbb{F}_2^n \times \mathbb{F}_2^m$ for which the function F under study (for instance a block cipher or a cryptographic hash function) admits more solutions $x \in \mathbb{F}_2^n$ to the corresponding differential equation $F(x) + F(x + \Delta^{\text{in}}) = \Delta^{\text{out}}$ than the expected number when F is picked uniformly at random. Soon after the invention of differential cryptanalysis, optimal functions resisting differential attacks were defined [Nyb94]. These functions are known as *Almost Perfect Nonlinear* (APN) and

correspond to functions F for which each equation $F(x) + F(x + \Delta^{\text{in}}) = \Delta^{\text{out}}$ has at most 2 solutions when $\Delta^{\text{in}} \neq 0$. They are indeed optimal since some of these equations must admit solutions and the minimal number is necessarily 2 since solutions come by pair: x is a solution if and only if $x + \Delta^{\text{in}}$ is also a solution.

When this optimal property was exhibited, first examples [Nyb94, BD94, Dob99a, Dob99b, Dob01] were also identified and a line of research [BCL08, BCL09a, BCL09b, BCV20, BBMM11, BCC+20, BHK20, ZKL+22, LZLQ22, ZP13, Tan19, Göl23, Göl22, GK25, CLV22, LK23] started to look for more exotic examples. This trend is important for advancing in the theoretical understanding of the combinatorial properties of APN functions, but also for assessing whether there exist structural properties that are inherent to all of them. The most well-known open problems in the field are for example the so-called *Big APN Problem* [BDKM09] which is to find an APN bijection $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ when n is even, or the certainly-as-hard problem to find an APN function which is inequivalent to any quadratic function and any power mapping. Solving such problems, even when n is small, would be a big step forward in our understanding of these objects.

For both problems, we currently know a single example of such functions, both in dimension 6: the Dillon permutation [BDMW10] is the only known APN bijection (up to equivalence) in even dimension and the Brinkmann-Leander-Edel-Pott cubic [EP09, BL08] the only known function that is inequivalent to any quadratic function and any power mapping. Apart from them, the two problems for $n = 6$ as for $n > 6$ remain open.

Interestingly, these examples were identified by *somehow* modifying APN functions known at that time in a way that is known to preserve the APN property, and ensuring that the newly-obtained function was either bijective or inequivalent to a quadratic function.

In the case of the Dillon permutation, the function was found by inspecting the CCZ-equivalence class of a function known as the Kim mapping [BDMW10]. CCZ equivalence [CCZ98] is the more general equivalence relation of vectorial Boolean functions known so far that preserves the APN property. However, just like the less general *extended-affine (EA) equivalence* relation, it does not preserve bijectivity. Dillon and his co-authors thus had to detect the presence of such a bijection in the class and to build the associated *admissible* mapping to modify the graph of the Kim mapping into the graph of an APN permutation. Since then, the theoretical foundations of this technique have been clearly understood [CP19, CL22] and progresses have been made on algorithmic solutions and their implementations, for instance to detect the presence of a bijection within a class or to travel across all EA-equivalence classes within a given CCZ-equivalence class. However progresses are still needed since current algorithms for instance return multiple representatives per extended-affine equivalence class, which inevitably slows the traversal.

Regarding the Brinkmann-Leander-Edel-Pott cubic, which was independently found by the two first and two last authors, the initial discovery by Edel & Pott was obtained by substituting a single coordinate of a known quadratic APN function into a suitable non-quadratic coordinate. This modification was designed to preserve the APN property but it also happened to leave the CCZ-equivalence class of the original function for another so-far unknown class that does not contain any quadratic representative. Yet, the substituted coordinates which enable to leave (or not) the initial CCZ-equivalence class are not understood. Nonetheless, this so-called *switching neighbours* method is EA-invariant but *a priori* not CCZ-invariant, and can thus be used in pair with the traversal of all EA-equivalence classes of a CCZ-equivalence class. However, contrary to the case of the method of Dillon *et al.*, the theoretical and practical aspects of this technique have, to the best of our knowledge and apart from [GLL25, GL26a, GL26b], received little attention from the community.

Yet, since the discoveries of these two examples, the list of the known (quadratic, non-bijective) APN functions in even dimension has really shot up. There now exist between 20 and 30 infinite families of CCZ-equivalence classes of APN functions based on polynomial descriptions of vectorial Boolean functions which can be instantiated over infinitely many fields $\mathbb{F}_{2^n}$. These families in particular provide some examples of APN over small fields, but more importantly, enabled to prove that the number of APN functions over $\mathbb{F}_2^n$ grows exponentially in n [KZ21, KZ22]. In parallel of these pen-and-paper approaches, many computer-assisted searches identified sporadic APN functions (*i.e* not included - yet - in more general constructions) in dimension $n \leqslant 10$ [YWL14, YP22, BBL21, BL22]. The latest one culminates with the new findings of over 3.8 millions of APN functions in dimension 8 [BLL+25]. These correspond to as many starting points for an exploration of the CCZ-equivalence classes of APN functions reachable by a combined usage of the switching neighbours method, together with the traversal of EA-equivalence classes.

Contributions. In this paper, we provide the tools needed to explore the complete set of APN functions reachable thanks to currently known techniques, and then perform a first exploration using them. To do so, we tackle the following two theoretical problems that were already mentioned.

1. An algorithm for traversing all EA-equivalent classes within a given CCZ-equivalence class should not return multiple representatives per EA-equivalence class, as is currently the case.

2. The switching neighbours of a given function should also be filtered out, in particular the ones that necessarily lead to a function CCZ- or EA-equivalent to the original one.

To achieve our first goal in the case of quadratic APN functions, we carefully investigate their sets of automorphisms, and show how to generate them very efficiently using the ortho-derivative. By making sure that the action of the automorphism group on the representatives of EA-equivalence classes is *completely* factored out, we provide a way to obtain a single representative of each EA-equivalence class. This in particular explains the observation made by Canteaut, Couvreur & Perrin [CCP22, Section 4.2] that the current algorithm "may return several representatives for a given EA-[equivalence] class".

Secondly, we provide the first detailed investigation of the set of the switching neighbours of a function. In particular, we show how an overwhelming majority of them are trivial in the sense that they provide functions that are EA-equivalent to the original one. We also explain how to get rid of those, that is, how to once again "factor out" the set of switching neighbours to study. Concretely, we divide by 2^{3n} the number of functions to explore and provide an algorithm to compute switching neighbours that is 2^n times faster than the state of the art. Incidentally, we also obtain new invariants of extended-affine equivalence using the resulting set of non-trivial switching neighbours. This in particular enables us to efficiently distinguish some Gold power mappings in dimension 7 that are known to be hard to distinguish.

Building on the previous theoretical results, we also provide efficient algorithmic solutions to explore the set of APN functions reachable by using the switching neighbours technique, as well as EA-partition traversals. We pushed highly-efficient and easy to re-use implementations of the corresponding algorithms to the `sboxU` [PAB+26] library, with a new release:

https://zenodo.org/records/23501624.

It allowed us to generate databases containing all known APN functions for $n = 6$ and $n = 7$ and compute all switching neighbours for these functions. For $n = 8$, our algorithms are efficient-enough to compute the EA-partitions and the switching neighbours of the quadratic functions allowing us to investigate the properties of more than one billion APN functions. This allows us to for instance generate 3833 new CCZ-equivalence classes in dimension 8, and also to replicate some previously-reported results [LSS12, Cal20], which is a crucial step for such experimental studies. Our improved algorithms further enable us to provide new insights regarding small dimensions (6 or 7), such as the number of EA-equivalent classes, or a description of the transition graph between them. We also provide, for the quadratic EA-classes only, some insight on the structure of the switching neighbours graphs for dimension 8, 9 and 10.

Finally, we provide tinySQL databases containing these EA-classes in dimension 6 and 7 and a set of invariants for each functions to allow for efficient invariant-based filtering for new functions. In dimension 8, we provide such database for quadratic functions, as well as some supplementary datasets containing the compressed 1.29 billions EA-classes in the quadratic classes. In dimension 9 and 10, we provide a similar database for the known quadratic functions.

These datasets were published on Zenodo with a small tutorial on how to use them:

https://zenodo.org/records/24006393.

The experimental code needed to replicate them and the data found in the paper were published on Github, as well a detailed tutorial on how to use it:

https://github.com/314gali/apn_ccz_sn_explotation.

Outline. In the following, Section 2 is dedicated to the necessary preliminary results and notations. Our contributions are then detailed in the next three sections. More precisely, Section 3 takes on the partition of the CCZ-equivalence class of functions into EA-equivalence classes, Section 4 on the

computation and structure of the set of switching neighbours. These two sections are each divided into theoretical and algorithmic subparts to reflect how a better theoretical understanding enables us to provide algorithmic improvements. Section 5 summarizes the experiment made available by the previously-established results, most of them being out of reach by state-of-the-art techniques before this paper. In this last section, we also discuss some implementations considerations, as well as our new invariants and their efficiency.

2 Preliminaries

The finite field with two elements is denoted by $\mathbb{F}_2$ and the n-dimensional space over $\mathbb{F}_2$ by $\mathbb{F}_2^n$. The standard dot product between two elements $x, y \in \mathbb{F}_2^n$ is denoted by $x \cdot y \coloneqq \sum_{i=1}^{n} x_i y_i$.

Vectorial Boolean functions. We study vectorial Boolean functions, that is, functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ for some n, m. A vectorial Boolean function can be seen as the collection of m Boolean functions $F_1, \dots F_m \colon \mathbb{F}_2^n \to \mathbb{F}_2$ which satisfies $F(x) = (F_1(x), \dots, F_m(x))$ for all $x \in \mathbb{F}_2^n$. The unique Boolean functions $F_1, \dots F_m$ are the *coordinates* of F while we refer to their linear combinations as the *components* of F. More precisely, for $\alpha = (\alpha_1, \dots, \alpha_n) \in \mathbb{F}_2^n$, the component $\alpha \cdot F$ is the function defined by $\alpha \cdot F \coloneqq \sum_{i=1}^{n} \alpha_i F_i$. Each coordinate can be represented as a unique polynomial over $\mathbb{F}_2[X_1, \dots, X_n]/(X_i^2 + X_i, 1 \leqslant i \leqslant n)$ known as its *algebraic normal form* (ANF). The *algebraic degree* of a Boolean function is the degree of its ANF while the algebraic degree of a vectorial Boolean function is the maximum over the degrees of its coordinates. A vectorial Boolean function F of degree exactly 1 is called *affine* (or *linear* when $F(0) = 0$). This of course refers to $\mathbb{F}_2$-linearity. A function F of algebraic degree exactly two is called *quadratic*.

We usually reserve uppercase notations for vectorial Boolean functions such as F, G and lowercase notations to Boolean functions such as $f, g \colon \mathbb{F}_2^n \to \mathbb{F}_2$. For affine functions, we reserve uppercase notation such as A, B, C for functions over $\mathbb{F}_2^n$, while we prefer calligraphic notation such as $\mathcal{A}, \mathcal{B}, \mathcal{C}$ for affine mappings defined from $(\mathbb{F}_2^n)^2$ to itself. The linear part of an affine mapping $\mathcal{A}$ (resp. A) is denoted by $\mathcal{L}_\mathcal{A}$ (resp. L_A), where the subscript may be omitted if the context is clear. Given a linear mapping L over $\mathbb{F}_2^n$, we denote by $L^\top$ the adjoint of L, that is, the unique linear mapping that satisfies $\forall x, y \in \mathbb{F}_2^n, x \cdot Ly = L^\top x \cdot y$.

Studying the APN property of a function F sums up to the study of its *derivatives*. For any $\Delta \in \mathbb{F}_2^n$, the derivative along Δ of F is the function $D_\Delta F$ defined by $D_\Delta F(x) = F(x) + F(x + \Delta)$ for any $x \in \mathbb{F}_2^n$. A function is therefore APN if and only if, for any $\Delta \neq 0$, the image set $D_\Delta F(\mathbb{F}_2^n)$ is of cardinality 2^{n-1}.

Equivalence relations. Vectorial Boolean functions are often classified and studied up to equivalence. The most general equivalence relation in use today is actually an equivalence relation on graphs of functions. The graph of a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is the subset Γ_F of $(\mathbb{F}_2^n)^2$ defined by:

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

Two such functions F and G are *CCZ-equivalent* [CCZ98] if there exists an affine permutation $\mathcal{A}$ over $\mathbb{F}_2^{2n}$ such that $\Gamma_G = \mathcal{A}(\Gamma_F)$. The affine mappings such that $\mathcal{A}(\Gamma_F)$ is indeed the graph of another function are called *admissible* with respect to F, and are in practice very rare.

Extended-Affine equivalence (or in short EA-equivalence) is a particular case of CCZ-equivalence that corresponds to the case where the matrix of $\mathcal{L}_\mathcal{A}$ is a lower-triangular block matrix, *e.g.* of the form

$$\mathcal{L}_\mathcal{A} = \begin{pmatrix} A & 0 \\ C & B \end{pmatrix} , \tag{1}$$

where A, B and C are $n \times n$ matrices. Note that $\mathcal{A}$ being bijective, both A and B are of maximum rank n and thus invertible. Conversely, any invertible affine mapping $\mathcal{A}$ satisfying Equation (1) is an admissible mapping for any function F. Following [CP19], affine bijections satisfying Equation (1) are called *EA-mappings*. For the sake of brevity, an equivalence class with respect to CCZ-equivalence (resp. EA-equivalence) will be called a CCZ-class (resp. EA-class), as in [CP19].

In the context of this paper, the reason for studying functions up to CCZ- or EA-equivalence is that these relations preserve the APN property. In other words, if F, G are two CCZ-equivalent (resp. EA-equivalent) functions, then F is APN if and only if G is APN. An APN function is therefore considered "new" if it lies outside already-exhibited CCZ-equivalence classes. As already mentioned,

the vast majority of the known CCZ-equivalence classes of APN function contain some quadratic representative. In the following, a function is said to be *CCZ-quadratic* [EP09, Definition 2] if it is CCZ-equivalent to a quadratic function.

Automorphims groups. An affine bijection $\mathcal{A}$ satisfying $\mathcal{A}(\Gamma_F) = \Gamma_F$ is an *automorphism* of (the graph of) F. The group of automorphisms of F is denoted by $\mathrm{Aut}(F)$, while the subgroup of automorphisms that are EA-mappings is denoted by $\mathrm{Aut}_{\mathrm{EA}}(F)$. The subgroup $\mathrm{Aut}_{\mathrm{lin}}(F)$ corresponds to the subgroup of block-diagonal automorphisms of the form $\mathrm{diag}(A, B) \coloneqq \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix}$.

Finally, for any $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$, it is straight-forward to observe that the derivative D_aF is affine if and only if F satisfies $\mathcal{A}(\mathcal{G}_F) = \mathcal{G}_F$ where:

$$\mathcal{L}_{\mathcal{A}} = \begin{pmatrix} \mathrm{Id} & 0 \\ C & \mathrm{Id} \end{pmatrix} \quad \text{and} \quad \mathcal{A}(0) = \begin{pmatrix} a \\ b \end{pmatrix},$$

for some C and b. If this holds, then C is actually the matrix of the linear part of D_aF, while $b = D_aF(0)$. We thus define the group $\mathrm{Aut}_{\Delta}(F) < \mathrm{Aut}_{\mathrm{EA}}(F) < \mathrm{Aut}(F)$ as the subgroup derived from affine derivatives, that is:

$$\mathrm{Aut}_{\Delta}(F) \coloneqq \left\{ \mathcal{A} \in \mathrm{Aut}(F) \ : \ \mathcal{L}_{\mathcal{A}} = \begin{pmatrix} \mathrm{Id} & 0 \\ \star & \mathrm{Id} \end{pmatrix} \right\}.$$

Walsh transform. Occasionally, we will need to use the well-known Walsh transform. Given a function $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^m$, the Walsh coefficient of F associated to $(\alpha, \beta) \in \mathbb{F}_2^n \times \mathbb{F}_2^m$ is defined by:

$$W_F(\alpha, \beta) \coloneqq \sum_{x \in \mathbb{F}_2^n} (-1)^{\alpha \cdot x + \beta \cdot F(x)}.$$

We will also rely on the so-called set of Walsh zeroes of F, which is the set $\mathcal{Z}_F \subset \mathbb{F}_2^n \times \mathbb{F}_2^m$ defined by:

$$\mathcal{Z}_F \coloneqq \{(0,0)\} \cup W_F^{-1}(\{0\}) \ = \ \big\{(0,0)\big\} \cup \big\{(\alpha, \beta) \in \mathbb{F}_2^{2n} \ : \ W_F(\alpha, \beta) = 0\big\}.$$

In particular, it is easy to see that, for any $\alpha \neq 0$, $W_F(\alpha, 0) = \sum_{x \in \mathbb{F}_2^n} (-1)^{\alpha \cdot x} = 0$ since the function $x \mapsto \alpha \cdot x$ is a balanced function. This implies that for any function F, it holds that $\mathbb{F}_2^n \times \{0\} \subset \mathcal{Z}_F$. In the following, we denote by $\mathcal{V}$ the set $\mathcal{V} \coloneqq \mathbb{F}_2^n \times \{0\}$. We also define Σ_F to be the set of n-dimensional subspaces in $\mathcal{Z}_F$ (so that $\mathcal{V} \in \Sigma_F$ for any F).

The case $n = m$. In the following, for the sake of clarity, we choose to state all of our results for functions $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$, that is for the specific case where $n = m$, which is, in the context of this paper, the most interesting case. We nonetheless note that most of the following results can easily be extended to the case where $n \neq m$.

3 Partition of a CCZ-equivalence class into EA-equivalence classes

As already mentioned, EA-equivalence is a particular case of CCZ-equivalence. This implies that any CCZ-class is actually the union of disjoint EA-classes. This nested structure also provides a natural equivalence relation on the admissible mappings with respect to F.

Definition 1 (EA-similarity [CP19, Definition 14 & Lemma 9])**.** Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a function and $\mathcal{A}$ and $\mathcal{A}'$ be admissible mappings with respect to F. Let G, G' be the functions defined by $\Gamma_G = \mathcal{A}(\Gamma_F)$ and $\Gamma_{G'} = \mathcal{A}'(\Gamma_F)$. Then $\mathcal{A}$ and $\mathcal{A}'$ are said to be EA-similar if G and G' are EA-equivalent.

By construction, EA-similarity is an equivalence relation over the set of admissible mappings with respect to F. We can also naturally characterize EA-similarity in terms of F, or more precisely in terms of its automorphisms, as described in Theorem 1 and illustrated in Figure 1.

Theorem 1. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a function and $\mathcal{A}$ and $\mathcal{A}'$ be admissible mappings with respect to F. Let G, G' be the functions defined by $\Gamma_G = \mathcal{A}(\Gamma_F)$ and $\Gamma_{G'} = \mathcal{A}'(\Gamma_F)$. Then G and G' are EA-equivalent if and only if there exists an EA-mapping $\mathcal{B}$ such that $\mathcal{A}'^{-1} \circ \mathcal{B} \circ \mathcal{A} \in \mathrm{Aut}(F)$.*

Proof. By definition, the functions G and G' are EA-equivalent if and only if there exists an EA-mapping $\mathcal{B}$ such that $\mathcal{B}(\Gamma_G) = \Gamma_{G'}$. However, $\mathcal{B}$ satisfies such an equality if and only if $\mathcal{B} \circ \mathcal{A}(\Gamma_F) = \mathcal{A}'(\Gamma_F)$, or equivalently if and only if $\mathcal{A}'^{-1} \circ \mathcal{B} \circ \mathcal{A}(\Gamma_F) = \Gamma_F$. □

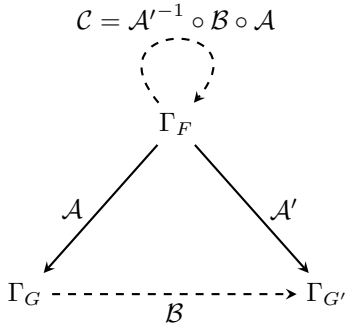

Figure 1: Graphical representation of Theorem 1.

In order to travel across all EA-classes in the CCZ-class of F, it is therefore sufficient to consider a single admissible mapping in each EA-similarity class.

3.1 The number of EA-equivalence classes in a CCZ-equivalence class

In [CP19], Canteaut & Perrin characterize CCZ-equivalence as the composition of EA-equivalence together with the so-called twist-equivalence. In particular, they rely on the following spectral characterization of admissible mappings, involving $\mathcal{V} = \mathbb{F}_2^n \times \{0\}$.

Proposition 1 ([CP19, Lemma 1 & Theorem 1])**.** *Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ and let $\mathcal{A}$ be an affine bijection over $(\mathbb{F}_2^n)^2$ with linear part $\mathcal{L}$. Then:*

(i) *$\mathcal{A}$ is admissible for F if and only if $\mathcal{L}^\top(\mathcal{V}) \subset \mathcal{Z}_F$.*

(ii) *$\mathcal{A}$ is an EA-mapping if and only if $\mathcal{V}$ is $\mathcal{L}^\top$-invariant,* i.e. *$\mathcal{L}^\top(\mathcal{V}) = \mathcal{V}$.*

Remark 1. As already mentioned, for any function F it holds that $\mathcal{V} \subset \mathcal{Z}_F$. The combination of the two previous items thus provides yet-another proof of why all EA-mappings are admissible for all functions.

Based on Theorem 1, we can refine [CP19, Lemma 3].

Proposition 2. *Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$, $\mathcal{A}, \mathcal{A}'$ be admissible mappings with respect to F and G, G' the functions defined by $\Gamma_G = \mathcal{A}(\Gamma_F)$ and $\Gamma_{G'} = \mathcal{A}'(\Gamma_F)$. Let $V = \mathcal{L}_{\mathcal{A}}^\top(\mathcal{V})$ and $V' = \mathcal{L}_{\mathcal{A}'}^\top(\mathcal{V})$. Then G and G' are EA-equivalent if and only if there exists $\mathcal{C} \in \mathrm{Aut}(F)$ such that $\mathcal{L}_{\mathcal{C}}^\top(V') = V$.*

Proof. By Theorem 1, G and G' are EA-equivalent if and only if there exists an EA-mapping $\mathcal{B}$ such that $\mathcal{A}'^{-1} \circ \mathcal{B} \circ \mathcal{A} \in \mathrm{Aut}(F)$. This is equivalent to the existence of an automorphism $\mathcal{C} \in \mathrm{Aut}(F)$ such that $\mathcal{A}'\mathcal{C}\mathcal{A}^{-1}$ is an EA-mapping. By Proposition 1, this equivalently means that $\mathcal{L}_{\mathcal{A}'\mathcal{C}\mathcal{A}^{-1}}^\top(\mathcal{V}) = \mathcal{V} \iff \mathcal{L}_{\mathcal{A}^{-1}}^\top \mathcal{L}_{\mathcal{C}}^\top \mathcal{L}_{\mathcal{A}'}^\top(\mathcal{V}) = \mathcal{V} \iff \mathcal{L}_{\mathcal{C}}^\top(V') = V$. □

The case described in [CP19, Lemma 3] corresponds to the sufficient condition where $V = V'$. In that case, as $\mathrm{Id} \in \mathrm{Aut}(F)$, and as any subspace V is invariant under Id, G and G' are necessarily EA-equivalent. The authors also provide the example of involutive mappings to prove that their condition was not necessary. In that case if $\mathcal{A} = \mathrm{Id}$ and $\mathcal{A}' = \begin{pmatrix} 0 & \mathrm{Id} \\ \mathrm{Id} & 0 \end{pmatrix}$, it is clear that $\mathcal{A}(\Gamma_F) = \Gamma_F = \mathcal{A}'(\Gamma_F)$, despite the fact that $\mathcal{L}_{\mathcal{A}}(\mathcal{V}) = \mathcal{V} \neq \mathcal{V}^\top = \mathcal{L}_{\mathcal{A}'}(\mathcal{V})$. However, as F is involutive, $\mathcal{B} = \begin{pmatrix} 0 & \mathrm{Id} \\ \mathrm{Id} & 0 \end{pmatrix} \in \mathrm{Aut}(F)$ and $\mathcal{B}(\mathcal{V}) = \mathcal{V}^\top$, which is the reason why $\mathcal{A}$ and $\mathcal{A}'$ still maps F to EA-equivalent mappings.

More importantly, Proposition 2 points out that we are interested in the action of the group $\mathrm{Lin}(\mathrm{Aut}(F))$ (made of the linear parts of automorphisms in $\mathrm{Aut}(F)$) on the set Σ_F of n-dimensional

subspaces in $\mathcal{Z}_F$. We define it as follows:

$$\begin{aligned}\rho\colon \mathrm{Lin}(\mathrm{Aut}(F)) \times \Sigma_F &\to \Sigma_F \\ (\mathcal{L}, V) &\mapsto \mathcal{L}^\top(V).\end{aligned} \tag{2}$$

Because a mapping $\mathcal{A}$ is admissible if and only if its linear part $\mathcal{L}_\mathcal{A}$ is, and since the functions G and G' defined by $\mathcal{A}(\Gamma_F) = \Gamma_G$ and $\mathcal{L}_\mathcal{A}(\Gamma_F) = \Gamma_{G'}$ are naturally EA-equivalent, the study of ρ is indeed sufficient to understand the partition of a CCZ-class into EA-classes. By construction and according to Proposition 2, the number of EA-classes in the CCZ-class of F is the number of orbits for this action. By using Burnside's lemma, we obtain the following refinement of [CP19, Theorem 4].

Theorem 2 (Number of EA-equivalence classes in a CCZ-equivalence class)**.** *Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$. Let e_F be the number of EA-equivalence classes in the CCZ-class of F. Then:*

$$e_F = \frac{1}{|\mathrm{Lin}(\mathrm{Aut}(F))|} \sum_{\mathcal{L} \in \mathrm{Lin}(\mathrm{Aut}(F))} |\mathrm{Fix}(\mathcal{L})|\,,$$

where $\mathrm{Fix}(\mathcal{L})$ is the set of n-dimensional subspaces of Σ_F that are invariant under $\mathcal{L}$. In particular it holds that:

$$\frac{|\Sigma_F|}{|\mathrm{Lin}(\mathrm{Aut}(F))|} \leqslant e_F \leqslant |\Sigma_F|\,.$$

The lower bound is obtained by only considering $\mathrm{Id} \in \mathrm{Lin}(\mathrm{Aut}(F))$ in the sum, while the upper bound is obtained by using $|\mathrm{Fix}(\mathcal{L})| \leqslant |\Sigma_F|$ for all $\mathcal{L} \in \mathrm{Lin}(\mathrm{Aut}(F))$. This matches the bounds given in [CP19, Theorem 4] whenever $|\mathrm{Lin}(\mathrm{Aut}(F))| = |\mathrm{Aut}(F)|$ which, in the study of APN functions, is the only interesting case. Indeed, the condition $|\mathrm{Lin}(\mathrm{Aut}(F))| \neq |\mathrm{Aut}(F)|$ is equivalent to F having a constant derivative $D_\Delta F$ for some $\Delta \neq 0$.

3.2 The particular case of quadratic APN functions

If $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a quadratic APN function, then partitioning its CCZ-class turns out to be feasible in practice. To do so, we rely on the fact that the automorphisms of quadratic APN functions are necessarily EA-mappings. This is recalled in Section 3.2.1. We also use the fact (recalled in Section 3.2.2) that the study of $\mathrm{Aut}_{\mathrm{EA}}(F)$ can be reduced to study of $\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$ where π_F is the *ortho-derivative* of F.

3.2.1 CCZ- and EA-equivalence between quadratic APN functions

It is well-known that two quadratic APN functions are CCZ-equivalent if and only if they are EA-equivalent [Yos12]. As observed by Kaspers & Zhou [KZ21, Lemma 2.4] the results of Yoshiara [Yos12] and the ones of Dempwolff & Edel [DE14] are actually more precise: if $\mathcal{A}(\Gamma_F) = \Gamma_G$ where F, G are both quadratic APN functions, then $\mathcal{A}$ is *necessarily* an EA-mapping.

Proposition 3 (CCZ equivalence of quadratic APN mappings)**.** *Let $n \geqslant 4$. Let $F, G\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ be quadratic APN mappings. Then:*

(i) *[KZ21, Lemma 2.4] any automorphism of F is an EA-mapping:* $\mathrm{Aut}(F) = \mathrm{Aut}_{\mathrm{EA}}(F)$*, and*

(ii) *if $\mathcal{A}$ is an admissible mapping satisfying $\mathcal{A}(\Gamma_F) = \Gamma_G$, then $\mathcal{A}$ is an EA-mapping.*

Sketch of Proof. Let us assume that $\mathcal{A}(\Gamma_F) = \Gamma_G$. In that case, $\mathrm{Aut}(F) = \mathcal{A}^{-1}\mathrm{Aut}(G)\mathcal{A}$. Because F is quadratic, $\mathrm{Aut}(F)$ admits a (2-elementary regular Abelian) subgroup, namely $\mathrm{Aut}_\Delta(F)$ which corresponds to the group of 2^n affine derivatives of F. Similarly, it holds that $\mathcal{A}^{-1}\mathrm{Aut}_\Delta(G)\mathcal{A} < \mathrm{Aut}(F)$ is another 2-elementary regular Abelian subgroup of $\mathrm{Aut}(F)$.

The fact that F and G are EA-equivalent [Yos12] comes from the fact that the subgroups $\mathrm{Aut}_\Delta(F)$ and $\mathcal{A}^{-1}\mathrm{Aut}_\Delta(G)\mathcal{A}$ are necessarily conjugate in $\mathrm{Aut}(F)$, see for instance [DE14, Theorems 3.11 & 3.12]. The fact that $\mathcal{A}$ is an EA-mapping comes from the fact that $\mathrm{Aut}_\Delta(F)$ is actually a normal subgroup of $\mathrm{Aut}(F)$, see [DE14, Theorems 4.10], which implies that $\mathcal{A}^{-1}\mathrm{Aut}_\Delta(G)\mathcal{A} = \mathrm{Aut}_\Delta(F)$. We then proceed just as in the proof of [DE14, Theorem 3.12] by considering the set of points that are fixed by all elements in $\mathrm{Aut}_\Delta(F)$, that we denote by $S_F = \left\{(x, y) \in (\mathbb{F}_2^n)^2 : \mathcal{B}(x, y) = (x, y), \forall \mathcal{B} \in \mathrm{Aut}_\Delta(F)\right\}$. It is clear that $S_F = \{0\} \times \mathbb{F}_2^n = S_G$. But the equality $\mathcal{A}^{-1}\mathrm{Aut}_\Delta(G)\mathcal{A} = \mathrm{Aut}_\Delta(F)$ also shows that $S_F = \mathcal{A}(S_G)$, which proves that $\mathcal{A}$ preserves $\{0\} \times \mathbb{F}_2^n$ and is therefore an EA-mapping. □

Remark 2. Note that Item **(ii)** can be reformulated as follows : given a quadratic APN function F and an admissible mapping $\mathcal{A}$, the function G satisfying $\mathcal{A}(\Gamma_F) = \Gamma_G$ is quadratic if and only if $\mathcal{A}$ is an EA-mapping. In other words, the orbit of $\mathcal{V}$ by the action ρ is of cardinality 1. This is clear since by Item **(i)** above the automorphism group of F is only made of EA-mappings which stabilize $\mathcal{V}$. In light of Proposition 2, this means that the space $V = \mathcal{V}$ cannot be sent onto any other space V' by any automorphism of F. Stated otherwise, any admissible mapping $\mathcal{A}$ w.r.t F such that $\mathcal{L}^{\top}_{\mathcal{A}}(\mathcal{V}) \neq \mathcal{V}$ (*i.e.* $\mathcal{A}$ is not an EA-mapping as per Proposition 1) necessarily provides a "non-trivial" CCZ-equivalent mapping, *i.e.* a mapping that is CCZ-equivalent but not EA-inequivalent to F.

These refined results provides an immediate answer to a question left open by Canteaut & Perrin regarding the degree of the functions obtained by applying a 1-twist to a quadratic function. We recall that a function F is t-twistable if the mapping $\mathcal{T}_t$ defined below is admissible with respect to F:

$$\mathcal{T}_t = \begin{pmatrix} 0 & 0 & \mathrm{Id}_t & 0 \\ 0 & \mathrm{Id}_{n-t} & 0 & 0 \\ \mathrm{Id}_t & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathrm{Id}_{n-t} \end{pmatrix}. \tag{3}$$

Corollary 1. *Let $n \geqslant 4$. Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN mapping. Then:*

(i) *[Yos12] There exists a single EA-class with quadratic mappings in the CCZ-class of F.*

(ii) *Let G be non-trivially t-twist equivalent to F. Then $\deg(G) \geqslant 3$. In particular, if G is 1-twist equivalent to F then $\deg(G) = 3$.*

(iii) *There exists a function of degree 3 which is CCZ-equivalent to F.*

(iv) *F is not t-twist self-equivalent. In particular F is not involutive.*

Proof. Item **(i)** is a direct corollary of the previous proposition. As shown by Equation 3, a twist-equivalence is a particular kind of CCZ-equivalence which is *not* an EA-equivalence. By the second item of Proposition 3, a function which is twist-equivalent to F must have degree at least 3. The 1-twist case is a consequence of the fact that a function derived from a quadratic function by a 1-twist has degree at most 3 [CP19, Section 6.3.1]. Item **(iii)** is implied by the previous point, together with the fact that a quadratic APN function is necessarily 1-twistable [CP19, Corollary 2]. Item **(iv)** is a consequence of **(ii)**. □

Note that a result similar to Items **(ii)** and **(iii)** was recently proven [JKK25, Theorem 6.7] with an explicit formula for a CCZ-equivalent function of degree 3.

The previous results can also be expressed in terms of Walsh zeroes. As detailed in the following proposition, if F is CCZ-quadratic (*i.e.* CCZ-equivalent to a quadratic function), then a *single* n-dimensional subspace of Walsh zeroes allows to move from F to any of its quadratic representative.

Proposition 4 (Admissible mappings from a CCZ-quadratic function to a quadratic function)**.** *Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a CCZ-quadratic APN function, Q a quadratic function and $\mathcal{A}$ an admissible mapping satisfying $\mathcal{A}(\Gamma_F) = \Gamma_Q$. Let $\mathcal{B}$ be an admissible mapping satisfying $\mathcal{B}(\Gamma_F) = \Gamma_G$ for some G. Then the function G is quadratic if and only if $\mathcal{L}^{\top}_{\mathcal{B}}(\mathcal{V}) = \mathcal{L}^{\top}_{\mathcal{A}}(\mathcal{V})$.*

Proof. This is a consequence of Proposition 2 and Proposition 3. Indeed, G is quadratic if and only if it is EA-equivalent to Q. This holds if and only if there exists $\mathcal{C} \in \mathrm{Aut}(F) = \mathcal{A}\mathrm{Aut}(Q)\mathcal{A}^{-1}$ such that $\mathcal{L}^{\top}_{\mathcal{C}}\mathcal{L}^{\top}_{\mathcal{A}}(\mathcal{V}) = \mathcal{L}^{\top}_{\mathcal{B}}(\mathcal{V})$ which can be simplified into $\mathcal{L}^{\top}_{\mathcal{A}}(\mathcal{V}) = \mathcal{L}^{\top}_{\mathcal{B}}(\mathcal{V})$ using the fact that $\mathrm{Aut}(F)$ is made of EA-mappings. □

Remark 3. The equality $\mathcal{L}^{\top}_{\mathcal{B}}(\mathcal{V}) = \mathcal{L}^{\top}_{\mathcal{A}}(\mathcal{V})$ can also be restated as the fact $\mathcal{A}\mathcal{B}^{-1}$ is an EA-mapping since it equivalently means that $\mathcal{V}$ is an invariant subspace of $\mathcal{L}^{\top}_{A B^{-1}}$ which is the definition of an EA-mapping given in Proposition 1.

The previous proposition allows us to generalize Remark 2 to not only quadratic but also CCZ-quadratic functions.

Corollary 2. *Let F be a CCZ-quadratic APN function. Then:*

(i) *there exists a unique vector space $V_F \subset \mathcal{Z}_F$ which satisfies the following equivalence: for any admissible mapping $\mathcal{B}$ with respect to F, the function G defined by $\mathcal{B}(\Gamma_F) = \Gamma_G$ is quadratic if and only if $\mathcal{L}_{\mathcal{B}}^{\top}(\mathcal{V}) = V_F$.*

(ii) *Furthermore, if F is quadratic, then $V_F = \mathcal{V}$.*

As of today, computing V_F is as hard as finding an explicit CCZ-equivalence relation between F and a quadratic function Q since we do not know any cheaper way to test the CCZ-quadratic property of a function. However in practice all known CCZ-quadratic functions are obtained the other way around, that is through the traversal of EA-classes of a quadratic function. In that case, V_F is obtained with no additional cost. Indeed, the functions CCZ-equivalent to a quadratic function Q are explored and built by selecting a Walsh zeroes subspace W of Q and computing a function F satisfying $V_F = W$.

In general, testing whether CCZ-quadratic functions are CCZ-*inequivalent* can be effectively done by proving that the corresponding quadratic representatives are EA-inequivalent, for instance by means of invariants, such as the Walsh and differential spectra of their ortho-derivatives. However, if the invariants are equal, no definite answer is obtained. In this context, the previous proposition provides an effective CCZ-equivalence test based on the computation of the automorphisms group of one of the two quadratic representatives. It is particularly efficient when a single CCZ-quadratic function F needs to be tested against a family of other functions. In that case, the automorphism group $\mathrm{Aut}(Q)$ of some quadratic function Q such that $Q \sim_{\mathrm{CCZ}} F$ can be precomputed once and for all and the CCZ-equivalence tests can be efficiently carried out using the next proposition.

Proposition 5 (CCZ-equivalence test for CCZ-quadratic functions)**.** *Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a CCZ-quadratic APN function satisfying $\mathcal{A}(\Gamma_F) = \Gamma_Q$ for some quadratic mapping Q and admissible mapping $\mathcal{A}$ (so that $\mathcal{L}_{\mathcal{A}}^{\top}(\mathcal{V}) = V_F$). Let G be a CCZ-quadratic APN function and V_G be the vector space defined in Proposition 4. Then F and G are CCZ-equivalent if and only if there exists $\mathcal{C} \in \mathrm{Aut}(Q)$ such that $\mathcal{L}_{\mathcal{C}}^{\top}(V_F) = V_G$.*

Proof. The functions F and G are CCZ-equivalent if and only if G and Q are CCZ-equivalent. If G and Q are CCZ-equivalent then let us consider $\mathcal{B}$ satisfying $\mathcal{B}(\Gamma_G) = \Gamma_Q$, so that $\mathcal{L}_{\mathcal{B}}^{\top}(\mathcal{V}) = V_G$. The mapping $\mathcal{B}\mathcal{A}^{-1}$ is thus an automorphism of Q and the linear part of its transpose sends V_F onto V_G. Conversely, if $\mathcal{C} \in \mathrm{Aut}(Q)$, then $\mathcal{L}_{\mathcal{C}\mathcal{A}^{-1}}^{\top}(\mathcal{V}) = V_G$, so by Corollary 2, $\mathcal{C}\mathcal{A}^{-1}$ is an admissible mapping between G and Q. □

As we can see by comparing Proposition 5 and Proposition 2, a CCZ-equivalence test between two CCZ-quadratic functions sums up to an EA-equivalence test (between the quadratic representatives of F and G). Similarly, when applied to CCZ-quadratic functions, we obtain the following restatement of Proposition 2 that is depicted in Figure 2. Contrary to the previous proposition, note that $\mathcal{A}$ maps Γ_Q to Γ_F and not the other way around, meaning that $\mathcal{L}_{\mathcal{A}}^{\top}(\mathcal{V})$ is a priori not equal to V_F in this context.

Proposition 6 (EA-equivalence test for CCZ-quadratic functions)**.** *Let $F, G\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ be CCZ-quadratic functions and Q, Q' be quadratic functions satisfying $\mathcal{A}(\Gamma_Q) = \Gamma_F$ and $\mathcal{A}'(\Gamma_{Q'}) = \Gamma_G$, for some admissible mappings $\mathcal{A}, \mathcal{A}'$. Let us assume that Q and Q' are EA-equivalent and $\mathcal{D}$ be an admissible mapping satisfying $\mathcal{D}(\Gamma_Q) = \Gamma_{Q'}$. Then F and G are EA-equivalent if and only if there exists $\mathcal{C} \in \mathrm{Aut}(Q)$ such that $\mathcal{L}_{C}^{\top}\left(\mathcal{L}_{\mathcal{A}}^{\top}(\mathcal{V})\right) = \mathcal{L}_{\mathcal{D}}^{\top}\left(\mathcal{L}_{\mathcal{A}'}^{\top}(\mathcal{V})\right)$.*

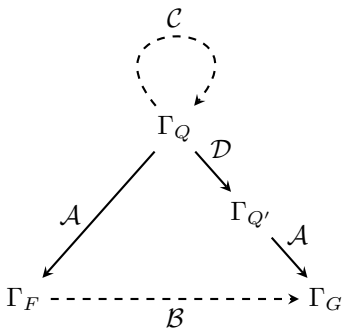

Figure 2: Graphical representation of Proposition 6.

The similarities between Propositions 2, 5 and 6 will enable us to adopt a common strategy to handle the algorithmic aspects of these tests in Section 3.3.

3.2.2 EA-equivalence of quadratic APN functions & ortho-derivatives

In the case of quadratic APN functions, it is known that the study $\mathrm{Aut}_{\mathrm{EA}}(F)$ can be reduced to the study of the group of *extended-linear* automorphisms of F (when $F(0) = 0$) [KZ21, Proposition 2.3], or to the study of *linear* automorphisms of the so-called *ortho-derivative* of F. In this section, we recall and combine these results to obtain a more fine-grained method for computing the automorphism group of quadratic APN functions. An extended-linear mapping (or EL-mapping) is an EA-mapping $\mathcal{A}$ satisfying $\mathcal{A}(0) = 0$. We define the subgroup $\mathrm{Aut}_{\mathrm{EL}}(F)$ in a similar manner as $\mathrm{Aut}_{\mathrm{EA}}(F)$ is defined.

Definition 2 (Ortho-derivative [CCP22, Definition 36])**.** Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN function. The *ortho-derivative* of F is the function $\pi_F \colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ such that $\pi_F(0) = 0$ and such that for any $a \in \mathbb{F}_2^n \setminus \{0\}$, $\pi_F(a)$ is the only non-zero element satisfying:

$$\pi_F(a) \cdot \big(F(x+a) + F(x) + F(a) + F(0)\big) \;=\; 0 \quad \forall\, x \in \mathbb{F}_2^n\,. \tag{4}$$

Lemma 1. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a quadratic APN function. Let A, B be affine bijections, and C be a linear function mapping $\mathbb{F}_2^n$ to itself. Let $G = B \circ F \circ A + C$. Then:*

(i) *[CCP22, Proposition 36] $\pi_G = L_B^\top \circ \pi_F \circ L_A^{-1}$.*

(ii) *In particular, if $F = B \circ F \circ A + C$ then $\pi_F = L_B^\top \circ \pi_F \circ L_A^{-1}$.*

Lemma 2 ([KZ21, Prop. 2.3])**.** *Let F be a quadratic function with $F(0) = 0$. Then it holds that*

$$\mathrm{Aut}_{\mathrm{EA}}(F) = \mathrm{Aut}_{\Delta}(F) \rtimes \mathrm{Aut}_{\mathrm{EL}}(F)\,.$$

We can observe that the second point of Lemma 1 can be reformulated as follows: for any linear bijections A, B such that $\begin{pmatrix} A & 0 \\ \star & B \end{pmatrix} \in \mathrm{Aut}_{\mathrm{EL}}(F)$, it holds that $\begin{pmatrix} A^{-1} & 0 \\ 0 & B^\top \end{pmatrix} \in \mathrm{Aut}_{\mathrm{lin}}(\pi_F)$. It should also be noted that the value $\star$ is in that case uniquely determined by the extended-linear self-equivalence relation and is equal to $BF + FA$. Combining and reformulating the previous lemmas, we thus derive the following important result on the structure of the automorphism group of quadratic APN functions.

Proposition 7 (Structure of $\mathrm{Aut}_{\mathrm{EA}}(F)$ for APN quadratic functions)**.** *Let $F \colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an APN quadratic function with $F(0) = 0$. Then it holds that*

(i) $\mathrm{Aut}_{\mathrm{EL}}(F)$ *is isomorphic to a* subgroup *of* $\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$*. More precisely let $\mathbb{G}$ be the subgroup of* $\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$ *defined by:*

$$\mathbb{G} = \left\{\mathrm{diag}(A, B) \in \mathrm{Aut}_{\mathrm{lin}}(\pi_F) \;:\; B^\top \circ F + F \circ A^{-1} \text{ is linear}\right\}.$$

Then the following homomorphism $\Phi \colon \mathbb{G} \to \mathrm{Aut}_{\mathrm{EL}}(F)$ is a group isomorphism:

$$\begin{array}{rccc} \Phi\colon & \mathbb{G} & \longrightarrow & \mathrm{Aut}_{\mathrm{EL}}(F) \\ & \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix} & \longmapsto & \begin{pmatrix} A^{-1} & 0 \\ B^\top \circ F + F \circ A^{-1} & B^\top \end{pmatrix}. \end{array}$$

(ii) *In particular,* $\mathrm{Aut}_{\mathrm{EA}}(F) \simeq \mathrm{Aut}_{\Delta}(F) \rtimes \mathbb{G}$ *and if $n \geqslant 4$, then* $\mathrm{Aut}(F) \simeq \mathrm{Aut}_{\Delta}(F) \rtimes \mathbb{G}$.

Proof. First, we observe that $\mathbb{G}$ is a subgroup of $\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$. Indeed, $\mathbb{G}$ naturally contains $\mathrm{diag}(\mathrm{Id}, \mathrm{Id})$ since $F + F = 0$ is linear and if both $B_1^\top F + FA_1^{-1}$ and $B_2^\top F + FA_2^{-1}$ are linear for some linear maps A_1, A_2, B_1, B_2 then, so is $(B_1B_2)^\top F + F(A_1A_2)^{-1}$ since

$$\begin{aligned} (B_1B_2)^\top F + F(A_1A_2)^{-1} &= B_2^\top B_1^\top F + (B_2^\top FA_1^{-1} + B_2^\top FA_1^{-1}) + FA_2^{-1}A_1^{-1} \\ &= B_2^\top (B_1^\top F + FA_1^{-1}) + (B_2^\top F + FA_2^{-1})A_1^{-1}\,. \end{aligned}$$

By the same identity, Φ is well-defined from $\mathbb{G}$ to $\mathrm{Aut}_{\mathrm{EL}}(F)$ and is injective by construction. The surjectivity of Φ is a direct consequence of Lemma 1 and the fact that if $\begin{pmatrix} A^{-1} & 0 \\ C & B^\top \end{pmatrix}$ is an element of $\mathrm{Aut}_{\mathrm{EL}}(F)$, then $C = B^\top F + FA^{-1}$. □

Algorithmic concerns. The algorithmic relevance of this property is twofold. First, it means that when F is APN and quadratic, then computing $\mathrm{Aut}(F)$ boils down to separately computing $\mathrm{Aut}_\Delta(F)$ and $\mathrm{Aut}_{\mathrm{EL}}(F)$ (which is 2^n times smaller than $\mathrm{Aut}_{\mathrm{EA}}(F)$). In fact, for most of the known (quadratic APN) functions the group $\mathrm{Aut}_{\mathrm{EL}}(F)$ is trivial.

Secondly, the structure of semi-direct product of $\mathrm{Aut}(F)$ when $n \geqslant 4$ (or of $\mathrm{Aut}_{\mathrm{EA}}(F)$ otherwise) allows us for simpler equality tests in the Walsh domain. Indeed, as it is suggested by Propositions 2, 5 and 6, given some subspaces V and V', searching for some $\mathcal{L} \in \mathrm{Aut}(F)$ such that $\mathcal{L}^\top(V) = V'$ can be done more efficiently by looking for collisions between two lists of n-dimensional spaces, namely between $\left(\mathcal{B}^\top(V)\right)_{\mathcal{B}\in\mathrm{Aut}_{\mathrm{EL}}(F)}$ and $\left(\mathcal{A}^{\top-1}(V')\right)_{\mathcal{A}\in\mathrm{Aut}_\Delta(F)}$. This can be efficiently done by for instance using hash tables. We detail how to use these properties in the following section.

Conjecture on the group structure. Finally, a natural corollary of Proposition 7 is that $\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$ and $\mathrm{Aut}_{\mathrm{EL}}(F)$ are isomorphic if and only if $B^\top F + FA^{-1}$ is linear for any $\mathrm{diag}(A, B) \in \mathrm{Aut}_{\mathrm{lin}}(\pi_F)$. In our experiments, this property holds for all the functions tested. Proving this fact or finding a counter example is left as open question.

Conjecture 1. *Let $n \geqslant 4$. Let F be a quadratic APN function with $F(0) = 0$. Then $\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$ and $\mathrm{Aut}_{\mathrm{EL}}(F)$ are always isomorphic. Equivalently, for any $\mathrm{diag}(A, B)$ in $\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$, the mapping $B^\top F + FA^{-1}$ is always linear.*

3.3 Computing one representative per EA-equivalence class in a CCZ-equivalence class

At the time of writing, many APN functions are known. Some of them lie in infinite families defined by polynomial forms, which in particular can be instantiated for smaller field sizes. On the other hand, in recent years, computer-aided searches have yielded literal tens of thousands of other APN functions: all EA-classes containing quadratic APN functions are known in $\mathbb{F}_{64}$ (there are 13 [BDMW10]), and in $\mathbb{F}_{128}$ (there are 488 [KI23]). Over $\mathbb{F}_{256}$, Beierle, Langevin, Leander, Polujan and Rasoolzadeh recently discovered more than 3.8 millions quadratic APN classes and conjecture there are even more [BLL+25].

Of course, all these functions come with their CCZ-classes which, as we established in Corollary 1, are non-trivial. It is now easy to go through said CCZ-class using e.g. `sboxU` [PAB+26] by generating all admissible mappings up to composition with EA-mappings. However, because of the existence of automorphisms, these admissible mappings may (and, in practice, very often are) EA-similar.

Computing the automorphism group of a quadratic APN function. Fortunately, in the case of quadratic APN functions, it is in fact possible to solve this problem. To this end, we first find all the automorphisms of a quadratic APN function F using the following approach.

Algorithm 1 Automorphism group of a quadratic APN function.

Input: A quadratic APN function F.
Output: $\mathrm{Aut}(F)$

1: Compute π_F.
2: Compute $\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$.
3: Initialize an empty set $\mathrm{Aut}_{\mathrm{EL}}(F)$.
4: **for** $\mathrm{diag}(A, B) \in \mathrm{Aut}_{\mathrm{lin}}(\pi_F)$ **do**
5: **if** $B^\top F + FA^{-1}$ is linear **then**
6: Add $\begin{pmatrix} A^{-1} & 0 \\ B^\top F + FA^{-1} & B^\top \end{pmatrix}$ to $\mathrm{Aut}_{\mathrm{EL}}(F)$.
7: **end if**
8: **end for**
9: Initialize an empty set $\mathrm{Aut}_\Delta(F)$.
10: **for** $a \in \mathbb{F}_2^n$ **do**
11: Add $\begin{pmatrix} \mathrm{Id} & 0 \\ D_aF + F(0) & \mathrm{Id} \end{pmatrix} + \begin{pmatrix} a \\ D_aF(0) \end{pmatrix}$ to $\mathrm{Aut}_{\mathrm{EL}}(F)$.
12: **end for**
13: **return** either $(\mathrm{Aut}_{\mathrm{EL}}(F), \mathrm{Aut}_\Delta(F))$ or $\mathrm{Aut}_\Delta(F) \rtimes \mathrm{Aut}_{\mathrm{EL}}(F)$ depending on the use-case.

The only non-trivial step of this algorithm is the computation of $\mathrm{Aut}_{\mathrm{EL}}(F)$. In practice, we rely on a straight-forward technique in the spirit of the work of Heggebakk & Kaleyski [HK22]. More specifically the difference distribution table (DDT) and the linear approximation table (LAT) of an ortho-derivative are not nearly as structured as the ones of a quadratic APN function. In particular its LAT, $\{\!\{W_{\pi_F}(\alpha,\beta)\}\!\}_{\alpha,\beta\in\mathbb{F}_2^n}$ admit a lot of distinct values and a linear-self equivalence $\mathrm{diag}(A,B)\in\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$ of π_F must map the partition of the preimages of W_{π_F} (two Walsh coefficients are in relation if and only if they have the same value) onto itself. This allows a depth-first search where at each stage a new image for A or B is fixed. The fact that the value of the Walsh coefficients are numerous imply that the number of guesses per vector of a given basis are limited, and so the width of the tree is limited. On the other hand because each new constraint provides others due to the linearity of A and B, useless branches can quickly be pruned, which limits the depth of most branches. We implemented such a computation in `sboxU`. The function `automorphisms_from_ortho_derivative` we added to `sboxU`[1] implements the whole process described in Algorithm 1.

Computing a system of EA-representatives in a CCZ-quadratic class. With $\mathrm{Aut}(F)$ being now computed, we proceed to obtain exactly one representative of each EA-class in the CCZ-class of F. To do so, we aim at keeping a single representative of the orbits for the action $\rho\colon \mathrm{Lin}(\mathrm{Aut}(F))\times\Sigma_F\to\Sigma_F$ of the group of linear parts of automorphisms $\mathrm{Lin}(\mathrm{Aut}(F))$ on the set Σ_F of n-dimensional subspaces of Walsh zeroes defined in Equation (2). From there, we build a set of admissible mappings which forms a complete set of representatives for the EA-similarity relation defined in Definition 1. This ensures that we obtain a single representative of each EA-class without missing any class. This strategy is summarized in Algorithm 2.

Algorithm 2 Generating a system of representatives of the EA-classes within a given CCZ-class.

Input: A function F and its automorphism group $\mathrm{Aut}(F)$.
Output: A system of representatives of the EA-classes in the CCZ-class of F.

1: Compute the set Σ_F of n-dimensional spaces of Walsh zeroes.
2: Compute $\mathcal{O}_1,\dots,\mathcal{O}_s$ the partition of Σ_F into orbits for the action $\rho\colon \mathrm{Lin}(\mathrm{Aut}(F))\times\Sigma_F\to\Sigma_F$.
3: Initialize an empty list `Sys_of_rep`.
4: **for** $1\leqslant i\leqslant s$ **do**
5: Select $V_i\in\mathcal{O}_i$.
6: Build an admissible mapping $\mathcal{L}_i$ satisfying $\mathcal{L}_i^{\top}(\mathcal{V})=V_i$.
7: Compute the look-up table of the function G_i defined by $\mathcal{L}_i(\Gamma_F)=\Gamma_G$.
8: Append G_i to `Sys_of_rep`.
9: **end for**
10: **return** `Sys_of_rep`.

Steps 1, 6, and 7 were already implemented in `sboxU` [BPT19, CP19]. The search for n-dimensional subspaces is a depth-first search made by keeping echeloned bases of vector spaces being built. This avoids having in the end multiple bases of the same space [BPT19]. Regarding steps 6 and 7, they are immediate. Indeed for a given i, according to Item **(ii)** of Proposition 1, it is sufficient to fix a basis $a_1,\dots,a_n$ of $\mathcal{V}$ and a basis $b_1,\dots,b_n$ of V_i, to freely complete them into bases of $\mathbb{F}_2^{2n}$ $a_1,\dots,a_{2n}$ and $b_1,\dots,b_{2n}$ and to build the unique linear mapping mapping a_j to b_j for each j to obtain $\mathcal{L}_i^{\top}$.

Finally, only Step 2 remains to be handled. Recovering the orbits for the action ρ is actually equivalent to recovering the connected components of the undirected graph with Σ_F as set of vertices and where an edge is drawn between V and V' whenever there exists $\mathcal{A}\in\mathrm{Aut}(F)$ such that $\mathcal{L}_{\mathcal{A}}^{\top}(V)=V'$. The naive approach costs around $|\mathrm{Aut}(F)|\times|\Sigma_F|$ since for each $\mathcal{A}\in\mathrm{Aut}(F)$, we need to compute the image of each $V\in\Sigma_F$.

However, this does not take into account any knowledge on the structure of $\mathrm{Aut}(F)$ and the complexity can *e.g.* be reduced to $r\times\Sigma_F$ when a generating set of $\mathrm{Aut}(F)$ of cardinality r is known. Because of Lemma 1, finding such a generating set reduces to finding a generating set of $\mathrm{Aut}_{\mathrm{EL}}(F)$. When it is not trivial, there are many examples where $\mathrm{Aut}_{\mathrm{EL}}(F)$ remains highly-structured and even cyclic in some cases. Yet, in the vast majority of the cases, or at least experimentally, it happens that $\mathrm{Aut}_{\mathrm{EL}}(F)$ is trivial. In these cases, it holds that $\mathrm{Aut}(F)=\mathrm{Aut}_{\Delta}(F)$, and is in particular 2-elementary Abelian (of order 2^n when F is quadratic). In that case, the generating set is given by a choice of

[1] https://github.com/lpp-crypto/sboxU/blob/v2.1/sboxU/apn/cython_functions.pyx#L89

basis of $\mathbb{F}_2^n$, and this improved approach provides a gain of $\frac{2^n}{n}$ compared to the naive approach. This corresponds to the least possible gain of the improved technique on the naive one. Indeed, the size $d(\mathbb{G})$ of a minimal generating set of a group $\mathbb{G}$ necessarily satisfies $d(\mathbb{G}) \leqslant \log_2(\mathbb{G})$ since adding a generator g multiplies the size of the generated (sub)group by $\mathrm{ord}(g) \geqslant 2$. In any cases, this represents a significative improvement over the naive strategy that we numerically evaluate in Section 5.1.1.

Both the naive approach and the improved approach when $\mathrm{Aut}(F) = \mathrm{Aut}_\Delta(F)$ have been implemented into `sboxU`. More precisely, the function `enumerate_ea_classes_apn_quadratic` [2], on the input of a quadratic APN function, returns a unique representative of each EA-class in the CCZ-class of a quadratic function.

Finally, it should be noted that this traversal of EA-classes is completely independent of the property (*e.g.* APN or quadratic) of the function F under study and only depends on our ability to efficiently compute its automorphism group. This can for instance be applied to non-quadratic APN monomials in small dimensions or to the Brinkmann-Leander-Edel-Pott cubic.

An efficient EA-equivalence test. As suggested by Proposition 6, with the knowledge of $\mathrm{Aut}(F)$, we can provide an efficient EA-equivalence test when $\mathrm{Aut}(F)$ is small.

Algorithm 3 EA-equivalence test for CCZ-quadratic functions.

Input: Two CCZ-quadratic functions F, G.
Output: `True` if $F \sim_{\mathrm{EA}} G$, `False` otherwise.

1: Compute any $\mathcal{A}$ admissible and Q quadratic such that $Q \sim_{\mathrm{EA}} F$ and $\Gamma_Q = \mathcal{A}(\Gamma_F)$.
2: Compute any $\mathcal{A}'$ admissible and Q' quadratic such that $Q' \sim_{\mathrm{EA}} G$ and $\Gamma_{Q'} = \mathcal{A}'(\Gamma_G)$.
3: **if** Q and Q' are EA-inequivalent **then**
4: **return** `False`
5: **else**
6: Compute any $\mathcal{D}$ such that $\mathcal{D}(\Gamma_Q) = \Gamma_{Q'}$.
7: **end if**
8: Compute $\mathrm{Aut}(Q)$ using its ortho-derivative.
9: **for** $\mathcal{C} \in \mathrm{Aut}(Q)$ **do**
10: **if** $\mathcal{L}_C^\top\left(\mathcal{L}_{\mathcal{A}}^\top(\mathcal{V})\right) = \mathcal{L}_{\mathcal{D}}^\top\left(\mathcal{L}_{\mathcal{A}'}^\top(\mathcal{V})\right)$ **then**
11: **return** `True`
12: **end if**
13: **end for**
14: **return** `False`

The efficiency of this algorithm is conditioned by the efficiency of computing a quadratic representative in a CCZ-quadratic class, computing EA-mappings between two quadratic functions and computing the automorphism group of a quadratic function. As noted after Corollary 2, the first step is usually free. As per Lemma 1, the second step can be efficiently achieved by listing all linear relations $B \circ \pi_F = \pi_G \circ A$ between π_F and π_G and verifying whether $B^\top F + G \circ A^{-1}$ is affine. As explained in the previous paragraph, the last step greatly benefits from the knowledge of the generators of $\mathrm{Aut}(Q)$: the *for* loop (lines 9 to 13) can be replaced by the already mentioned collision search of such $\mathcal{C}$. We discuss the practical efficiency of this algorithm in Section 5.1.1. This algorithm has been implemented in sboxU as `are_ea_equivalent_from_quadratic_representative`[3]

Remark 4. Note that, as is, Algorithm 3 is not symmetrical: only the group $\mathrm{Aut}_{\mathrm{lin}}(Q)$ is computed. Hence, there is a clear gain putting the function with the smallest automorphism group as the F input.

4 Revisiting switching neighbours

With the exploration of a given CCZ-class being handled in the previous section, we now tackle the task of going outside it. Of course, when searching for APN functions, a natural idea is to leave a known CCZ-class and reaches another class which also contains APN functions. To the best of our knowledge, the only systematic technique to achieve this goal is the one based on switching neighbours introduced by Edel and Pott [EP09], and inspired from the so-called Dillon's switching construction mentioned in

[2] https://github.com/lpp-crypto/sboxU/blob/v2.1/sboxU/ccz/cython_functions.pyx#L212
[3] https://github.com/lpp-crypto/sboxU/blob/v2.1/sboxU/ccz/cython_functions.pyx#L423

[EP09, Section 3] and [BCL09a, Remark 2]. The authors used this strategy to great success, finding new classes of $6, 7$ and 8 bit APN functions. As we will show, this technique is somehow EA-invariant but not CCZ-equivalent: the CCZ-classes that can be reached from the EA-classes of a given CCZ-class are not necessarily the same. This implies that the switching method can in theory be used in pair with the exploration of EA-classes described before to investigate more CCZ-classes.

However, since the seminal work of [EP09], it seems that this the experimental aspects of this technique have received little attention, except from [GLL25, GL26a, GL26b].

The goal of this section is to clarify some of them. A first objective is to clarify what is the set of "interesting" switching neighbours and in particular identify and eliminate some "trivial" ones which necessarily lead to already known EA-classes. This serves our main goal which is of course to obtain a more efficient exploration algorithm, that we describe at the end of this section.

4.1 Definitions and Characterization

In the following, we only focus on the notion of switching neighbours in *the narrow sense* of Edel and Pott [EP09, Definition 5] and not the more general definition. For the sake of convenience, we refer to them as switching neighbours. In the following, we recall this narrow definition and introduce some important notations. Given a vectorial Boolean function $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$, a vector $u \in \mathbb{F}_2^n$ and a Boolean function $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$, we denote by $F + u \times f$ the function defined by:

$$F + u \times f(x) = \big((F_1 + u_1 f)(x), \ldots, (F_n + u_n f)(x)\big) .$$

Definition 3 (APN Switching Neighbours)**.** Let $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be two APN functions. The functions F and G are APN switching neighbours if there exists $u \in \mathbb{F}_2^n$ and a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ such that $F + G = u \times f$. For any $u \in \mathbb{F}_2^n$, the set $\mathsf{SN}_F(u)$ is the set of switching neighbours of F with respect to u, defined by:

$$\mathsf{SN}_F(u) \coloneqq \{G = F + u \times f \ \mid \ f\colon \mathbb{F}_2^n \to \mathbb{F}_2, \text{ s.t. } G \text{ is APN}\} .$$

The set of *switches* with respect to F and u is defined by:

$$\mathsf{sw}_F(u) \coloneqq \{f\colon \mathbb{F}_2^n \to \mathbb{F}_2 \ \mid \ F + u \times f \in \mathsf{SN}_F(u)\} .$$

We also define the *switching neighbourhood* SN_F of F, as the union of the sets $\mathsf{SN}_F(u)$, that is:

$$\mathsf{SN}_F \coloneqq \bigcup_{u \in \mathbb{F}_2^n} \mathsf{SN}_F(u) = \{G \ \mid \ \exists\, v \in \mathbb{F}_2^n, f\colon \mathbb{F}_2^n \to \mathbb{F}_2, \text{ s.t. } G = F + v \times f \text{ and } G \text{ is APN}\} .$$

The important difference between the set of Boolean functions $\mathsf{sw}_F(u)$ and the sets of *vectorial* Boolean functions $\mathsf{SN}_F(u)$ and SN_F are emphasized by the respective lower- and upper-cases.

Remark 5. The set SN_F is not an equivalence class. Indeed, the switching neighbour property is not transitive for distinct u and v. Edel and Pott [EP09, Definition 6] thus suggest to consider the smallest class containing CCZ-classes (or respectively EA-classes), and their switching neighbours. From time to time, we refer to these classes as CCZ-SN and EA-SN classes.

In their seminal paper, the authors also give a characterization for a function of the form $F + u \times f$ to be an APN switching neighbour. It is recalled in the following theorem.

Theorem 3 (Characterization of APN switching neighbours [EP09, Theorem 9])**.** *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an APN function. Let $u \in \mathbb{F}_2^n$, $u \neq 0$, and let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a Boolean function. The function $F + u \times f$ is an APN function (or equivalently $f \in \mathsf{sw}_F(u)$) if and only if:*

$$f(x) + f(x + a) + f(y) + f(y + a) = 0$$

for all $x, y, a \in \mathbb{F}_2^n$ which satisfy:

$$F(x) + F(x + a) + F(y) + F(y + a) = u. \tag{5}$$

This characterization provides an algorithm to compute the switching neighbours that we discuss and improve in the dedicated section. We now recall some simple facts on the switching neighbours construction that are directly deduced from this criterion.

Lemma 3. *Let $u \in \mathbb{F}_2^n$.*

(i) *The relation over functions from $\mathbb{F}_2^n$ to itself that is defined by $F \sim G \iff G \in \mathsf{SN}_F(u)$ is an equivalence relation.*

(ii) *The set $\mathsf{sw}_F(u)$ is a vector space.*

(iii) *The set $\mathsf{SN}_F(u)$ is an affine space.*

Proof. For fact **(i)**, reflexivity stems from the fact that $F = F + u \times 0$ is APN, symmetry stems from the fact that u is its own additive inverse over $\mathbb{F}_2^n$ and transitivity stems from the fact the set of Boolean functions is an additive group.

For fact **(ii)**, Theorem 3 states that if $f \in \mathsf{sw}_F(u)$ then for all $x, y, a \in \mathbb{F}_2^n$ such that $F(x) + F(x + a) + F(y) + F(y + a) = u$, the function f must satisfy $f(x) + f(x + a) + f(y) + f(y + a) = 0$. However, this condition is linear so if $f_1, f_2 \in \mathsf{sw}_F(u)$, then $f_1 + f_2 \in \mathsf{sw}_F(u)$. We can then easily verify that $\mathsf{sw}_F(u)$ is a vector space. Fact **(iii)** is then easily deduced from fact **(ii)**. □

4.2 Filtering switching neighbours

Since APN functions are studied up to EA- or CCZ-equivalence, there is a subset of the switching neighbours that we are not particularly interested in: the functions $F + u \times f$ that are equivalent to F.

Definition 4. Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an APN function, $u \in \mathbb{F}_2^n$ and $\mathrm{Class}_{\mathrm{CCZ}}(F)$ and $\mathrm{Class}_{\mathrm{EA}}(F)$ be respectively the CCZ-class and EA-class of F. We denote the set of switching neighbours of F with respect to u that are CCZ-equivalent to F by $\mathsf{SN}_F^{\mathrm{CCZ}}(u)$, namely:

$$\mathsf{SN}_F^{\mathrm{CCZ}}(u) = \mathsf{SN}_F(u) \cap \mathrm{Class}_{\mathrm{CCZ}}(F)\,.$$

Similarly, we denote by $\mathsf{sw}_F^{\mathrm{CCZ}}(u)$ the set of corresponding switches, defined by the following set equality:

$$\mathsf{SN}_F^{\mathrm{CCZ}}(u) = F + u \times \mathsf{sw}_F^{\mathrm{CCZ}}\,.$$

We similarly define the sets $\mathsf{SN}_F^{\mathrm{EA}}(u), \mathsf{sw}_F^{\mathrm{EA}}(u)$ by $\mathsf{SN}_F^{\mathrm{EA}}(u) = \mathsf{SN}_F(u) \cap \mathrm{Class}_{\mathrm{EA}}(F)$ and $\mathsf{SN}_F^{\mathrm{EA}}(u) = F + u \times \mathsf{sw}_F^{\mathrm{EA}}$. As in Definition 3, $\mathsf{SN}_F^{\mathrm{CCZ}}$ and $\mathsf{SN}_F^{\mathrm{EA}}$ are defined as the respective unions of all $\mathsf{SN}_F^{\mathrm{CCZ}}(u)$ and $\mathsf{SN}_F^{\mathrm{EA}}(u)$.

From the nested structure of CCZ and EA equivalences, it is clear that $\mathsf{SN}_F^{\mathrm{EA}} \subset \mathsf{SN}_F^{\mathrm{CCZ}}$. Further, in all of our experiments, we did see a clear difference between the two notions, which is hard to characterize in general. For instance, there are functions in dimension 6 for which $\mathsf{SN}_F^{\mathrm{EA}} = \mathsf{SN}_F^{\mathrm{CCZ}}$ like x^3 and some for which $\mathsf{SN}_F^{\mathrm{EA}} \neq \mathsf{SN}_F^{\mathrm{CCZ}}$. In practice and as already noticed by Edel and Pott [EP09, Remark 1.2], it seems hard to fully characterize the sets $\mathsf{SN}_F^{\mathrm{CCZ}}$ and $\mathsf{SN}_F^{\mathrm{EA}}$ even though they are exactly the sets that we would like to filter out in our exploration.

In the following, we instead provide a partial answer to this problem by identifying a subset of $\mathsf{SN}_F^{\mathrm{EA}}$ and $\mathsf{SN}_F^{\mathrm{CCZ}}$ which is easier to characterize and that appears for every function. In the following, we denote by $\mathrm{Aff}(n)$ the vector space of affine (or constant) Boolean functions from $\mathbb{F}_2^n$ to $\mathbb{F}_2$.

Proposition 8. *Let $n > 2$, let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an APN function and $u \in \mathbb{F}_2^n$. Then:*

(i) $\langle F_1, ..., F_n \rangle \cap \mathrm{Aff}(n) = \{0\}$,

(ii) $\mathrm{Aff}(n) \subset \mathsf{sw}_F^{EA}(u) \subset \mathsf{sw}_F(u)$ *,and*

(iii) $\langle F_1, ..., F_n \rangle \cap \mathsf{sw}_F(u) \subset \mathsf{sw}_F^{EA}(u)$. *Furthermore,* $\dim(\langle F_1, ..., F_n \rangle \cap \mathsf{sw}_F(u)) = n - 1$.

Proof. Fact **(i)** is equivalent to stating that F does not admit any affine component. This holds because $n > 2$ and F is APN. Indeed if is the case, up to EA-equivalence, we can assume that $F_n = 0$. In that case, the function $x \mapsto (F_1(x), \ldots, F_{n-1}(x))$ has balanced derivatives. It is therefore a bent function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^{n-1}$ which cannot exist since $n > 2$ [Nyb91], see *e.g.* [Car21, Proposition 161].

Fact **(ii)** stems from the fact that for any $f \in \mathrm{Aff}(n)$, $F + u \times f$ is EA-equivalent to F with $(A, B, C) = (\mathrm{Id}, \mathrm{Id}, u \times f)$.

Finally, for Fact **(iii)**, we first observe that $dim(\langle F_1, ..., F_n \rangle) = n$ for the same reason as for Fact **(i)**. Indeed, if it is not the case, then up to EA-equivalence, $x \mapsto (F_1(x), \ldots, F_{n-1}(x))$ would be perfect

nonlinear which is equivalent to being bent [Nyb91] and already proven to be impossible with such input/output dimensions. Then, using Theorem 3, we observe that for a fixed $u \neq 0$, f is in $\mathsf{sw}_F(u)$ if and only if for all $1 \leqslant i \leqslant n$, if $x, y, a \in \mathbb{F}_2^n$ satisfies $F_i(x) + F_i(x+a) + F_i(y) + F_i(y+a) = u_i$, then $f(x) + f(x+a) + f(y) + f(y+a) = 0$. For any $\lambda \in u^\perp$, by definition, $\lambda \cdot F$ is a Boolean function that satisfies $\lambda \cdot F(x) + \lambda \cdot F(x+a) + \lambda \cdot F(y) + \lambda \cdot F(y+a) = \lambda \cdot u = 0$. Let $v \notin u^\perp$. Then since $\mathbb{F}_2^n = \langle v \rangle \oplus u^\perp$ and $v \cdot F(x) + v \cdot F(x+a) + v \cdot F(y) + v \cdot F(y+a) = v \cdot u = 1$, it holds that:

$$\langle F_1, ..., F_n \rangle \cap \mathsf{sw}_F(u) = \{\lambda \cdot F : \ \lambda \in u^\perp\}$$

and is therefore of dimension $n-1$. Finally, the fact that $\langle F_1, ..., F_n \rangle \cap \mathsf{sw}_F(u) \subset \mathsf{sw}_F^{\mathrm{EA}}(u)$ is immediate by observing that each coordinate of $F + u \times (\lambda \cdot F)$ is a linear combination of the F_i's. □

As a direct corollary of the previous results, given a function F, there are at least 2^{2n} switches per direction u that provide a switching neighbour in the same EA-equivalence than F. The number $2n$ is the sum of the dimensions of $\mathrm{Aff}(n)$ (*i.e.* $n+1$) and $\langle F_1, ..., F_n \rangle \cap \mathsf{sw}_F(u)$ (*i.e.* $n-1$). In our context, since we are looking for APN functions outside known classes, these switches should be ignored. In particular and contrary to the set $\mathsf{SN}_F^{\mathrm{EA}}(u)$ for which no structure is known (and for which it is *e.g.* unclear whether it is a vector space or not), Proposition 8 suggests that $\mathrm{Aff}(n) \oplus \big(\langle F_1, ..., F_n \rangle \cap \mathsf{sw}_F(u)\big)$ is a large subspace of $\mathsf{SN}_F^{\mathrm{EA}}(u)$ that we will be able to efficiently compute and filter out from our exploration.

Definition 5 (Non-trivial switching class)**.** Let $n > 2$, let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an APN function and let $u \in \mathbb{F}_2^n \setminus \{0\}$. The non-trivial switching class $\mathsf{SN}_F^{nt}(u)$ of F with respect to u is an affine space of the form $F + u \times \mathsf{sw}_F^{nt}(u)$, where $\mathsf{sw}_F^{nt}(u)$ is the quotient of $\mathsf{sw}_F(u)$ by $\mathrm{Aff}(n) \oplus \langle F_1, ..., F_n \rangle \cap \mathsf{sw}_F(u)$. The union $\cup_{u \in \mathbb{F}_2^n \setminus \{0\}} \mathsf{SN}_F^{nt}(u)$ is called the non-trivial neighbourhood of F and denoted by SN_F^{nt}.

Remark 6. As the main focus of this paper is the efficient computation of switching neighbours, we do not compute the non-trivial switching classes as they are defined. Instead, we choose a basis of a supplementary of $\mathrm{Aff}(n) \oplus \langle F_1, ..., F_n \rangle \cap \mathsf{sw}_F(u)$ in $\mathsf{sw}_F(u)$. This basis is however not uniquely defined.

Note that a function F' in the non-trivial neighbourhood of F can still *a priori* be EA-equivalent to F. As an example, if F has algebraic degree 3 but admits a derivative along some $a \in \mathbb{F}_2^n$ whose coordinates are all linear except coordinate u, then $D_a F(x)$ can be decomposed as $D_a F(x) = L(x) + u \times q_a(x)$ for some linear function L and quadratic Boolean function q_a. It then holds that $F + u \times q_a$ does lie in the EA-class of F since $F(x) + u \times q_a(x) = F(x+a) + L(x)$. Furthermore, q_a is in that case neither a combination of coordinates of F nor an affine function.

Nonetheless, as suggested by our experimental results in Section 5, the non-trivial neighbourhood of an APN function is usually small and can be explored way more efficiently than the standard neighbourhood.

4.3 Switching neighbours and equivalence classes

Before diving into implementation considerations about the computation and exploration of the non-trivial neighbourhood, we study further the interactions of the switching technique with two equivalence relations, namely EA-equivalence and twist-equivalence. This will be again be very useful for detecting and avoiding some unnecessary explorations.

4.3.1 Swiching neighbours and EA-equivalence

An important first observation is the fact that the notion of switching neighbours is independent of the representative of the EA-class, in the following sense.

Proposition 9 (EA-invariance of switching neighbours)**.** *Let $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be two APN EA-equivalent functions. More precisely, let A, B, C be affine functions with A, B bijective such that $G = B \circ F \circ A + C$. Then it holds that:*

$$f \in \mathsf{sw}_F(u) \quad \iff \quad f \in \mathsf{sw}_G\big(B(u)\big) \circ A^{-1}\,.$$

Proof. Let $u \in \mathbb{F}_2^n \setminus \{0\}$, $f \in \mathsf{sw}_F(u)$ and $G = B \circ F \circ A + C$. Then, $B \circ (F + u \times f) \circ A + C$ is an APN function and we have:

$$B \circ (F + u \times f) \circ A + C = B \circ F \circ A + C + B(u) \times (f \circ A) = G + B(u) \times (f \circ A)\,.$$

Then, $f \circ A \in \mathsf{sw}_G(B(u))$, hence the result. □

Corollary 3. *Let $F, G : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be two APN EA-equivalent functions. Let H be an APN function. Then it holds that:*

$$\exists F' \in \mathsf{SN}_F, H \sim_{\mathrm{EA}} F' \quad \iff \quad \exists G' \in \mathsf{SN}_G, H \sim_{\mathrm{EA}} G' .$$

Proof. If H is EA-equivalent to any $F' \in \mathsf{SN}_F$, then there exist $u \in \mathbb{F}_2^n \setminus \{0\}$, $f \in \mathsf{sw}_F(u)$ and A', B' and C' such that:

$$H = B' \circ (F + u \times f) \circ A' + C' = B' \circ F \circ A' + C' + B'(u) \times (f \circ A') .$$

Since $B \circ F \circ A + C = G$ for some A, B, C, and since the previous proposition states that $f \in \mathsf{sw}_F(u) \iff f \circ A \in \mathsf{sw}_G(B(u))$, it holds that:

$$\begin{aligned} H &= B' \circ F \circ A' + C' + B'(u) \times (f \circ A') \\ &= B'(B^{-1}(G + C)A^{-1})A' + C' + B'(u) \times (f \circ A') \\ &= B'B^{-1}GA^{-1}A' + B'B^{-1}CA^{-1}A' + C' + B'B^{-1}B(u) \times (f \circ (AA^{-1}A')) \\ &= B'B^{-1}(G + B(u) \times (f \circ A))A^{-1}A' + B'B^{-1}CA^{-1}A' + C' . \end{aligned}$$

Hence, H is EA-equivalent to $G + B(u) \times (f \circ A)$ which belongs to SN_G. Since F and G play symmetric roles, we prove the second implication in a similar manner. □

Corollary 3 is thus of great significance as it states that the switching neighbours of a *single* representative of a given EA-class are enough to travel across all EA-classes in the neighbourhood of the original EA-class. Stated otherwise, the neighbourhoods of two EA-equivalent functions are, up to EA-equivalence, the same.

Incidentally, the previous proposition also translates to constraints on the automorphism group $\mathrm{Aut}_{\mathrm{EA}}(F)$ when F is APN.

Corollary 4. *Let F be an APN function. Let us assume that $F = B \circ F \circ A + C$ holds for some affine functions A, B, C with A, B bijective. Then for any $u \in \mathbb{F}_2^n$, it holds that $\mathsf{sw}_F(B(u)) = \mathsf{sw}_F(u) \circ A^{-1}$.*

Remark 7. This means that automorphisms of F are also constrained by their action on switching neighbours. However, as we will see in the next section, the information gathered from the neighbourhood SN_F can be very limited depending on F: in particular SN_F^{nt} is often empty when $n > 6$. This seems to limit the usage of the previous corollary in a generic strategy to efficiently compute automorphism groups. It could however be included on a case-by-case basis, for instance when SN_F^{nt} is non-empty and $\mathrm{Aut}(F)$ is non-trivial. On the other hand, if a function has a large automorphism group, it can lead to a profusion of switching neighbours for different values u: indeed a switching neighbour for u provides a switching neighbour for $B(u)$. This phenomenon is observed in particular in dimension 8 in Section 5 where we exhibit a large set of functions with non-trivial automorphism groups and which are switching neighbours ones from the others.

4.3.2 Switching neighbours and twist equivalence

Secondly, we observe that in some cases, the neighbourhood of two *twist-equivalent* functions F and G are also related one to the other.

Recall that two functions $F, G \colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ are said to be t-twistable for some $1 \leqslant t \leqslant n$ if there exists a family of bijections $(T_y \colon \mathbb{F}_2^t \to \mathbb{F}_2^t)_{y \in \mathbb{F}_2^{n-t}}$ and a family of functions $(U_x \colon \mathbb{F}_2^{n-t} \to \mathbb{F}_2^{n-t})_{x \in \mathbb{F}_2^t}$ such that F and G can respectively expressed as follows:

$$\forall (x, y) \in \mathbb{F}_2^t \times \mathbb{F}_2^{n-t}, \quad F(x, y) = (T_y(x), U_x(y)) \quad \text{and} \quad G(x, y) = (T_y^{-1}(x), U_{T_y^{-1}(x)}(y)) . \tag{6}$$

Such a decomposition of F and G is a so-called TU decomposition. We refer to [CP19, Section 4] for more details.

Proposition 10. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an APN function and G be the t-twist equivalent function. Let F and G be decomposed as in Equation (6). Then it holds that:*

$$f(x, y) \in \mathsf{sw}_F(0, u) \quad \iff \quad f(T_y^{-1}(x), y) \in \mathsf{sw}_G(0, u) .$$

Proof. A switching neighbour of F with direction $(0,u)$ can be written $F(x,y)+(0,u)\times f(x,y) = (T_y(x), U_x(y)+u\times f(x,y))$ which preserves the bijective property of $x\mapsto T_y(x)$ for each y. This implies that this switching neighbour is also t-twistable and its twist-equivalent function G' is then given by:

$$G'(x,y) = \left(T_y^{-1}(x), U_{T_y^{-1}(x)}(y) + u\times f(T_y^{-1}(x),y)\right) = G(x,y)+(0,u)\times f(T_y^{-1}(x),y)\,.$$

Since G is also t-twistable, the equivalence follows. □

This proposition immediately gives another filter for our exploration. Indeed, if we have already computed the switching neighbours of a function F, all the switching neighbours of the t-twist of F for points of the form $(0,u)$ with $u\in\mathbb{F}_2^{n-t}$ are t-twists of switching neighbours of F, and in particular in the CCZ-class of these neighbours. Stated otherwise, for these specific directions, computing a twist and switching neighbour are two operations which commute. For a given function F, if we start by computing its CCZ-class, we can then filter at most $1/2^t$ of directions for each t-twist with no additional cost.

Since any CCZ-equivalence can be expressed as the composition of an EA-mapping, a twist and another EA-mapping, we similarly obtain the following result.

Proposition 11. *Let F be an APN function. Let $\mathcal{A}=\begin{pmatrix}A & D\\ C & B\end{pmatrix}$ be an admissible mapping with respect to F and let G be the function defined by $\mathcal{A}(\Gamma_F)=\Gamma_G$. Let $u\in\mathbb{F}_2^n$ such that $L_D(u)=0$ and $L_B(u)\neq 0$. It holds that:*

$$\exists f\in \mathsf{sw}_F^{nt}(u) \quad\iff\quad \exists g\in \mathsf{sw}_G^{nt}(L_B(u))\,.$$

Again, this means that the switching neighbours for directions satisfying these properties are preserved by CCZ-equivalence, whereas the situation is more complex for other directions. Indeed, for these specific directions, we fail to exhibit an admissible mapping that transports the switching neighbours from F to G. However, we see experimentally that such switching neighbours exist.

4.4 Efficient computation of switching neighbours

In the previous sections, we theoretically identified that:

1. a large class of "trivial" switching neighbours can be factored out since they necessarily provide mappings that are EA-equivalent to the initial function (see Proposition 8),
2. it is sufficient to consider the neighbourhood of a single representative per EA-class (see Proposition 9), and
3. if we inspect the switches of all EA-classes in a given CCZ-class, then some switching directions of twist-equivalent functions can be avoided (see Proposition 10).

We now turn into making these observations efficient in practice. First and as already observed by Edel and Pott, Theorem 3 immediately gives an algorithm to compute the set $\mathsf{SN}_F(u)$ that we summarize in the following proposition.

Proposition 12. *(Rephrasing of [EP09]) Let $F:\mathbb{F}_2^n\to\mathbb{F}_2^n$ be an APN function. Let $u\in\mathbb{F}_2^n$, $u\neq 0$ and let $I_u=\{(x,y,a)\in(\mathbb{F}_2^n)^3 \mid F(x)+F(x+a)+F(y)+F(y+a)=u\}$ and let L_u be the set of vectors $L_u=\{\ell^\top \mid \ell\in\mathbb{F}_2^n, \exists(x,y,a)\in I_u, \mathrm{Supp}(\ell)=\{x,x+a,y,y+a\}\}$. Then $\mathsf{SN}_F(u)$ is exactly the right-kernel of any matrix M_u whose set of rows is L_u.*

Remark 8. Any matrix M_u from Proposition 12 is extremely sparse since each line is composed of at most 4 ones.

Computing the set SN_F thus boils down to computing the kernels of M_u. The naive way to do so is to go through all the triplets (x,y,a) and to add a row to M_u each time $F(x)+F(x+a)+F(y)+F(y+a)=u$. Doing so would require 2^{3n} in time and 2^{4n} in memory just to compute the matrices M_u. Then, computing the kernel of each M_u costs $|I_u|\times 2^{2n}$ using Gaussian elimination. To optimize this naive approach, one can notice that we do not need all the elements in I_u to compute the kernel of M_u, since $d_u = 2^n-\dim(\mathrm{Im}(M_u))$ independent rows suffice.

Computing switching neighbours with random sampling. To propose a more efficient algorithm, we first observe that if we did gather less than d_u independent samples to build a matrix M, we would not miss any switching neighbour since the computed kernel $\ker(M)$ would satisfy $\mathsf{sw}_F(u) = \ker(M_u) \subset \ker(M)$. The only obstacle is that if $\mathsf{sw}_F(u) \subsetneq \ker(M)$ then some of the solutions are not APN switching neighbours. However, assuming that $\ker(M_u)$ is not too big, such false-positive can just be discarded by a brute-force verification. This remark is important as it means that the only drawback of this probabilistic algorithm is to overestimate the size of the switching class $\mathsf{SN}_F(u)$, which can be filtered afterwards.

Then, one possible method to gather roughly d_u independent samples is to draw them at random. Assume we draw $n_{\text{rows}} < d_u$ rows at random to form a matrix M_u^{rand}. Then, if $\mathsf{rank}(M_u^{\text{rand}}) \neq \mathsf{rank}(M_u)$ and $\dim \ker(M_u^{\text{rand}})$ is not too big, we can exhaustively search for APN functions in $F + u \times \ker(M_u^{\text{rand}})$. We sum up this discussion by the following algorithm, that is implemented in `sboxU` as `switching_neighbours` [4].

Algorithm 4 Computation of Non-trivial Switching Neighbours of an APN function F.

Input: An APN function F and an integer n_{rows}.
Output: The list of $\mathsf{SN}_F^{nt}(u)$ for $u \in \mathbb{F}_{2^n}^*$.

1: While the matrices M_u do not have n_{rows} rows, we draw a new triplet (x, y, a) at random, and add the corresponding line to the corresponding M_u.
2: Add the $2n$ linear constraints to each M_u to remove the non-trivial switches from the kernels
3: For each u filter out the functions $F + u \times f$ for $f \in \ker(M_u)$ that are not APN.
4: **return** The list of $\mathsf{SN}_F^{nt}(u)$ for $u \in \mathbb{F}_{2^n}^*$.

Remark 9. (Parameter n_{rows}) The efficiency of the algorithm depends heavily on n_{rows}. Indeed, if the kernels $\ker(M_u)$ are too big, i.e. n_{rows} is too small, their enumeration can become costly. Conversely, n_{rows} being too big means that computing the kernels becomes costly. To choose the optimal parameters n_{rows} for our experiments, we relied on experimental benchmarks. Refer to Section 5 for more details.

5 Implementation considerations and experimental results

Based on the previous improvements on the partitioning of CCZ-classes into EA-classes and the computation of non-trivial switching neighbours, we describe our exploration experiments. All the claims of the following section are verifiable thanks to the code provided as supplementary material. Our code is based on the previously-presented algorithms and is implemented and available in `sboxU`. For dimension 6 and 7, the experiments and the data analysis can be reproduced by running few scripts on a laptop. For dimension 8, due to the large number of functions, we provide an experimental journal to help running the computation on a large machine, using parallel computation. We provide as much material as possible to encourage reproducibility: the data can be found on Zenodo (https://zenodo.org/records/24006393) and the scripts to reproduce our results on Github (https://github.com/314gali/apn_ccz_sn_explotation).

Our exploration methodology. Before diving into our experimental results, we clarify our main methodological choices. One problem that has mainly been eluded for now is the problem of testing if a function is new or more precisely, if it is equivalent to a known function. For CCZ-equivalence, an efficient algorithm providing a definite answer is lacking, though a recent preprint presents highly-encouraging progresses in this direction [KS26].

For our results, we adopt a standard method based on invariants. To do so, we use collections of state-of-the-art EA-invariants that we call a *mugshot* and which is itself a more fine-grained EA-invariant.

If F is quadratic, its mugshot is composed of the Walsh spectrum and the differential spectrum of its ortho-derivative. When F is not quadratic, it contains its absolute Walsh spectrum, its degree spectrum, its 4-sigma multiplicities and its thickness spectrum. As the goal of these mugshots is to facilitate the identification of new functions, they are included in all databases found in the supplementary material.

[4] https://github.com/lpp-crypto/sboxU/blob/v2.1/sboxU/apn/cython_functions.pyx#L289

5.1 Exploring the set of APN functions in practice

5.1.1 Taking advantage of the structure of $\mathrm{Aut}(F)$ and its generators

We briefly recap the efficiency of our partition algorithm and the improvements we get from the knowledge of the structure of $\mathrm{Aut}(F)$, either as a semi-direct product or even more precisely of its generators.

We make our numerical experiments in dimension 8 using quadratic functions that have a trivial automorphism group, i.e. of size 2^8 since $\mathrm{Aut}(F) = \mathrm{Aut}_\Delta(F)$ in that case, an intermediate group size (function 13615 in the database) and the largest known size (the cube). In these experiments, three modes are compared: the *standard* mode partitions the Walsh zeroes using the full group $\mathrm{Aut}(F)$, the *product* version partitions using the semi-direct structure of the group $\mathrm{Aut}(F) = \mathrm{Aut}_\Delta(F) \rtimes \mathrm{Aut}_{\mathrm{EL}}(F)$ and the *generator* one finds generators of both $\mathrm{Aut}_\Delta(F)$ and $\mathrm{Aut}_{\mathrm{EL}}(F)$ before partitioning. The reported runtimes are averaged over ten runs, and are only indicative. The goal of Table 1 is to be indicative of the improvements of each mode for the large experiments in dimension 8 we run in the rest of the section.

Table 1: Mean runtimes in second of the `enumerate_ea_classes` function for different automorphism-group cardinalities in dimension 8.

$\lvert\mathrm{Aut}(F)\rvert$	$\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$	Standard	Product	Generators
256	0.11	0.18	0.16	0.11
13,615	0.53	1.02	0.55	0.52
522,240	15.49	30.07	15.99	16.67

From the small to medium size automorphism group the improvement is clear: using the group structure is more efficient for partitioning and having generators improves the algorithm further, getting almost a factor 2 in the end. However, when $\mathrm{Aut}(F)$ becomes larger, the generator strategy becomes slightly less efficient. This is due to the fact that verifying that a set of elements is indeed a generator set gets quite costly when the group grows larger.

Remark 10. Note that since the cube is the function with the largest automorphism group, the structure of $\mathrm{Aut}(x^3)$ is well known and the generator algorithm can be improved with this knowledge. We did not use it in the benchmarks to highlight the generic aspect of our method.

Similarly, we report the efficiency of our EA-equivalence (Algorithm 3) using the same group structure and generators improvements. Note that we do not use the cube function here as there are only two EA-classes in its partition in dimension 8, meaning that an EA-equivalence between function of degree 3 in the CCZ-class of the cube can only be positive. To have negative tests we instead consider Function 9 in the database, which is the function with the largest automorphism group and strictly more than two EA-classes in it CCZ-class.

Table 2: Mean runtimes of the `ea_mapping_from_quadratic_representative` function for different automorphism-group cardinalities in dimension 8. The runtime for the EA-equivalence test success (left) and failure (right) are reported in second. The runtimes of each step of Algorithm 3 are also reported.

$\lvert\mathrm{Aut}(F)\rvert$	$\mathrm{Aut}_{\mathrm{lin}}(\pi_Q)$	$Q \sim_{\mathrm{CCZ}} F$	$Q \sim_{\mathrm{EA}} Q'$	Standard	Product	Generators
256	0.12	0.11	0.10	0.33/0.36	0.32/0.36	0.32/0.36
13,615	0.57	0.10	0.12	1.30/1.32	0.80/0.80	0.79/0.80
46,080	1.2	0.06	0.10	2.97/3.06	1.43/1.40	1.40/1.40

From the previous table, we clearly see that using the product or generator mode is similar. This is due to the fact that the product mode can directly look for collisions of the shape $\mathcal{L}_C^\top\left(\mathcal{L}_{\mathcal{A}}^\top(\mathcal{V})\right) = \mathcal{L}_{\mathcal{D}}^\top\left(\mathcal{L}_{\mathcal{A}'}^\top(\mathcal{V})\right)$ using tables for $\mathrm{Aut}_{\mathrm{lin}}(\pi_F)$ while the generator mode still has to compute entire orbits.

5.1.2 Exact partitioning of CCZ-classes into EA-classes

We can now state our experimental observations on the CCZ-partition and EA-partition in small dimensions.

Experimental results 1 (EA-partition in dimension 6)**.** *We have established the following facts:*

Number of known EA-classes among CCZ-quadratic classes. *The 13 quadratic APN functions of* $\mathbb{F}_{64}$ *contain, in total, 691 EA-classes in the union of their CCZ-classes. Though the technique is different, this matches the results of Calderini [Cal20].*

Number of known EA-classes. *Adding the only known class of APN functions that are not CCZ-quadratic functions yields 25 new EA-classes (as in [Cal20]), for a total of 716 EA-classes.*

Algebraic degrees of the EA-classes. *These 716 EA-classes have the following algebraic degrees: 13 are quadratic, 521 are cubic, and 182 are of degree 4. None are of degree 5. In particular, we recover the 534 classes containing quadratic and cubic functions mentioned by Langevin, Saygi & Saygi [LSS12].*

Sizes of the classes and of their automorphism groups. *The size of the automorphism groups and EA-partition size distributions are presented in Figure 3. As expected, the cube has the largest automorphism group followed by the second function in the Banff list. The smallest automorphism group comes from the Brinkmann-Leander-Edel-Pott non-quadratic function.* △

Remark 11. For the only non-quadratic function, we compute its automorphism group using a generic CCZ-equivalence algorithm, making it costly.

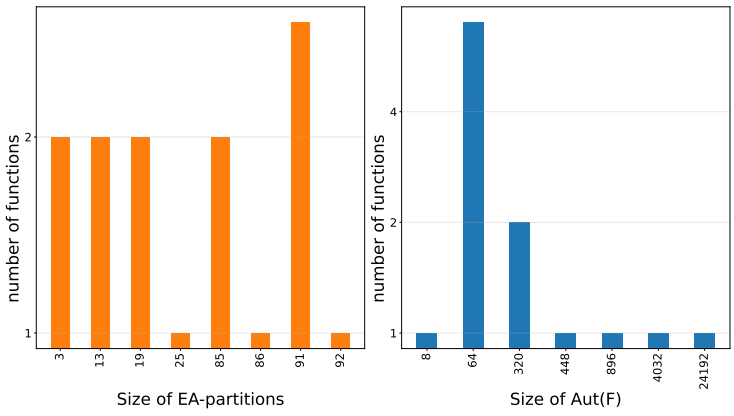

Figure 3: Distribution of the size of the automorphism group and size of the EA-partitions of the known APN functions in dimension 6.

Experimental results 2 (EA-partition in dimension 7)**.** *We have established the following facts:*

Number of known EA-classes among CCZ-quadratic classes. *The 488 quadratic APN functions of [KI23] over* $\mathbb{F}_{128}$ *contain, in total, 82,409 EA-classes in the union of their CCZ-classes. To the best of our knowledge, this is the first time this partition is computed.*

Algebraic degrees of the EA-classes. *These 82,409 EA-classes have the following algebraic degrees: 488 are quadratic, 60,302 are cubic, 21,619 are of degree 4. None are of degree 5 or 6.*

Sizes of the classes and of their automorphism groups. *The size of the automorphism groups and EA-partition size distributions are presented in Figure 4. The three quadratic APN monomials are almost the only functions without a trivial automorphism group.* △

Remark 12 (Number of known EA-classes in odd dimension). We do not report on the size of the partitions for the known non-quadratic functions as our algorithmic improvements are only for quadratic functions. Including them in our experiments in dimension 7 would largely increase the runtime of the database generation, as we would use generic CCZ-equivalence algorithms to compute the automorphism group.

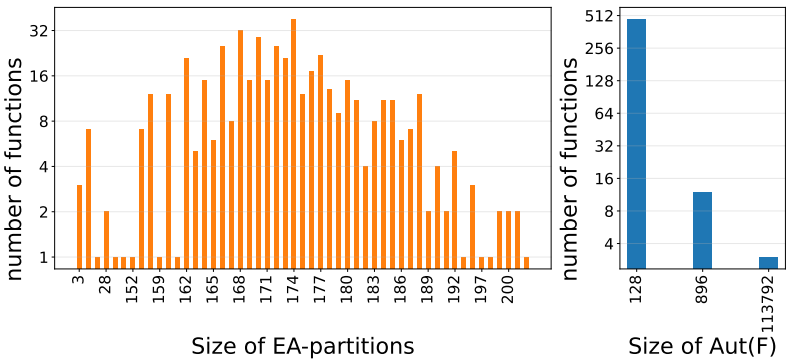

Figure 4: Distribution of the size of the automorphism group and size of the EA-partitions of the known quadratic APN functions in dimension 7.

Experimental results 3 (EA-partition in dimension 8)**.** *We have established the following facts:*

Number of known EA classes among CCZ-quadratic classes. *The 3,808,491 CCZ-quadratic classes $\mathbb{F}_{256}$ known before this paper from [YWL14, YP22, BBL21, BL22, BLL+25], contain, in total 1,290,800,505 EA-classes in the union of their CCZ-classes. This is the first time this partition is computed.*

Algebraic degrees of the EA-classes. *The degree distribution is the following: there are 968,994,078 cubic classes, 317,355,482 quartic classes and 642,454 quintic classes. Mostly notably, the common intuition that quadratic functions are easier to find is in complete contradiction with the distribution of degree we report: a random EA-class is of degree 3 or 4 with high probability. However, the functions of degree 5 are notably rare, even if some CCZ-quadratic classes have more than one EA-class of degree 5.*

Sizes of the classes and of their the automorphism groups. *The size of the automorphism groups and EA-partition size distributions are respectively presented in Figures 5 and 6.* △

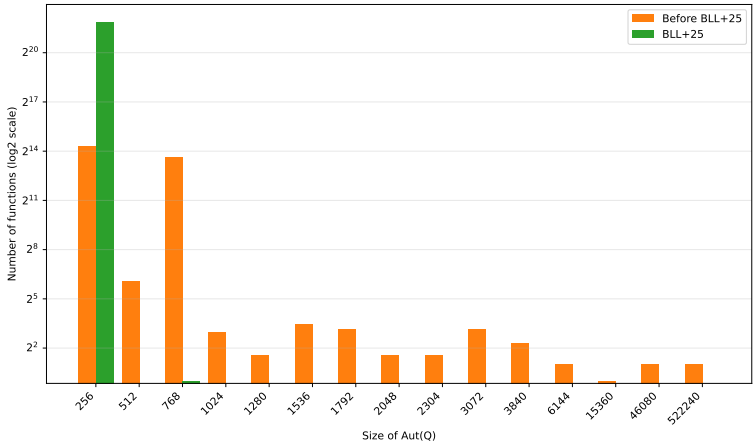

Figure 5: Distribution of the size of the automorphism group of the quadratic APN functions before and after the recent batch from [BLL+25].

Having such amount of data allows us to answer statistical questions about properties of random APN functions in dimension 8, such as what is the expected size of an EA-partition of the CCZ-class or of the automorphism group of a (known) APN function drawn at random. From Figures 5 and 6 and the information earlier, we clearly see that a random APN function among the known ones is of degree 3, has a trivial automorphism group and its CCZ-class partitioned into roughly 340 EA-classes. Further, if we compare the distributions of these values before the work of Beierle et al. [BLL+25]

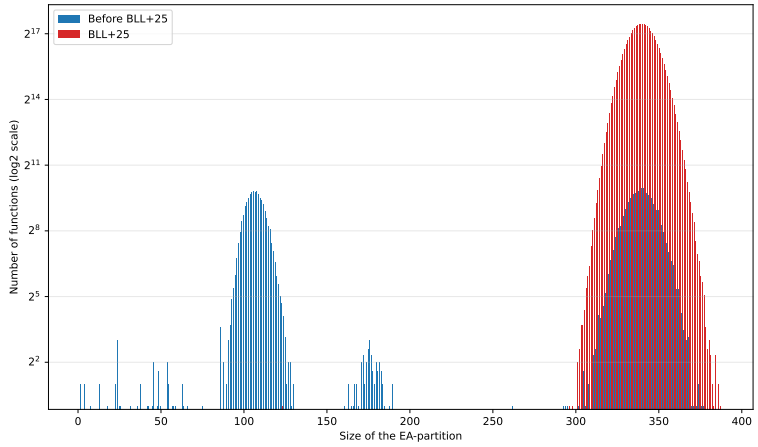

Figure 6: Distribution of the size of the EA-partition of the CCZ-class of the quadratic APN functions before and after [BLL+25]. The two distributions are juxtaposed on the same axis.

and after, we clearly see that the 3.8 millions functions they generated support the already-visible distribution of EA-partition size centered at 340. On the other hand, the functions generated before are more structured and exhibit automorphism groups of diverse sizes. In the set of Beierle et al, only one function has a non-trivial automorphism group.

We want to highlight the abundance of degree 3 functions in the dataset. Experimentally, we see that most quadratic functions have $255 = 2^8 - 1$ cubic functions in their EA-partition of CCZ-class. We also know that there are at least $2^8 - 1$ ways to make a 1-twist of the starting function and that by Corollary 1, their degree is 3. However, some of them could be EA-equivalent. This leads us to the following conjecture:

Conjecture 2. *There are at most $2^n - 1$ cubic EA-classes in the CCZ-class of a quadratic function. For an APN function drawn at random, the probability that there are exactly $2^n - 1$ such classes is close to 1.*

Remark 13. This probability cannot be 1 as there are pathologic functions like APN monomials for which the EA-partitions contains few classes. In these cases, the action of the highly non-trivial automorphism group "aggregate" the 1-twists into common orbits so that there are less than $2^n - 1$ EA-classes in the partition.

Experimental results 4 (EA-partition of the CCZ-class of the cube mapping)**.** *The CCZ-class of the function x^3 contains exactly 3 EA-classes for $n = 6$ and $n = 7$: itself, its inverse (whose components are all of degree 4), and another function obtained by a 1-twist. For $n = 8$, the partition contains only 2 classes: the cube and a 1-twist.* △

5.1.3 Exploring the switching neighbours of EA-classes

Using implementations of the previously-exposed algorithms, we computed for the first time the switching neighbours of the known APN functions in the literature in dimension 6, 7, 8, 9 and 10. This section summarizes the performances of these implementations and provides an analysis of these results. In dimension 8, this leads us to find 3833 new quadratic functions.

Parameters and Benchmarks. As we eluded in Section 4.4, we need to optimize the parameter n_{rows} for the switching neighbours computation. For a first estimation, if we consider our n_{rows} rows to be drawn at random in a 2^n-dimensional vector space over $\mathbb{F}_2$, it is known that the mean dimension of the span of these 2^n vectors should be *almost* of dimension 2^n: it boils down to computing the mean rank of a random $2^n \times 2^n$ matrix over $\mathbb{F}_2$. Furthermore, the situation where our matrix does not have full rank but the obtained $\dim\ker(M_u^{\text{rand}})$ is nonetheless small is sufficient, since we can enumerate the leftover functions to filter non-APN functions.

To refine this estimation, we experimentally make n_{rows} vary around 2^n to derive the optimal parameters. For most functions in dimension $n > 6$, we observe that $\dim\ker(M_u)$ is small and the

Table 3: Summary of the time and memory complexity of our `switching_neighbours` function for the optimal parameters n_{rows}. For each n, the parameters n_{rows} were derived experimentally.

Dimension	6	7	8	9	10
Optimal n_{rows}	64	128	270	620	1400
Mean Time Complexity (in sec)	0.015	0.06	0.34	1.80	11.42
Mean Memory Complexity (in MB)	0.6	2.7	9.0	43.9	249.0

enumeration is thus a trivial step. This implies that the row sampling and kernel computing steps are the most time-consuming. Furthermore, if n_{rows} get smaller that 2^n, a trade-off appears: in that case the kernels are large and the brute-force is costly, while the kernel computation is not. At larger dimensions, this can cause memory blow-up.

Overall, the experimental analysis confirms that $n_{\text{rows}} \approx 2^n$ is close to optimal for $n = 6$ and $n = 7$. For these dimensions, having large kernels $\ker(M_u^{\text{rand}})$ does not bring any complexity blow-ups in the brute-force phase as the tables are still small. However, starting $n = 8$, any large deflect in $\mathsf{rank}(M_u^{\text{rand}})$ leads to a large brute-force effort. The optimal parameters become slightly larger than 2^n. Table 3 summarize the time and memory complexity of the implementation of our algorithm using these optimal parameters.

The estimations of n_{rows} were made by drawing at random functions in the available databases. The dimension for which we have the most representative data is dimension 8 where we extensively checked for n_{rows} by drawing at random in the 1.29 billions functions. Most notably, the optimal n_{rows} and the runtime does not depend on the input function, not even its degree. The values reported in Table 3 are the midpoints of a range of optimal parameters: a slight variation around these values does not change significantly the time complexity of the algorithm.

Experimental Results. Having a representative of all known EA-classes means it becomes practical to test if a new function is actually new.

Experimental results 5 (Switching neighbours in dimension 6)**.** *We have established the following facts:*

No new APN neighbour. *By looking at switching neighbours of all known EA-classes in dimension $n = 6$, no new function can be found.*

Switching-Neighbour-CCZ graph. *We can exhaustively compute the Switching-Neighbour-CCZ graph for the 14 known classes (see Figure 7). In this graph, two CCZ classes are connected if there exists a switching neighbour going from one class to the other. Function 11 (in the reference database) plays a particular role: it is connected to all other functions in its connected component.*

Role of EA-classes in the Switching-Neighbour-CCZ graph. *In this graph, the CCZ-class of a quadratic function does not reach more CCZ-classes via switching neighbours than the quadratic function itself.* △

The previous result leads us to state the following conjecture shared by other authors, see *e.g.* [GL26b, Slide 3].

Conjecture 3. *All 6-bit APN functions are now known. They all fit in one of 691 EA-classes, those being further grouped into 14 CCZ-classes. In particular, there is no 6-bit APN function of degree 5 or 6.*

Experimental results 6 (Switching neighbours in dimension 7)**.** *We have established the following facts:*

No new APN neighbour. *By looking at switching neighbours of all 82,409 known EA-classes in dimension $n = 7$, no new function can be found.*

Switching-Neighbour-CCZ graph. *We can exhaustively compute the Switching-Neighbour-CCZ graph. The connected components are sparser: the size of the components range between 2 and 5 across 52 components.*

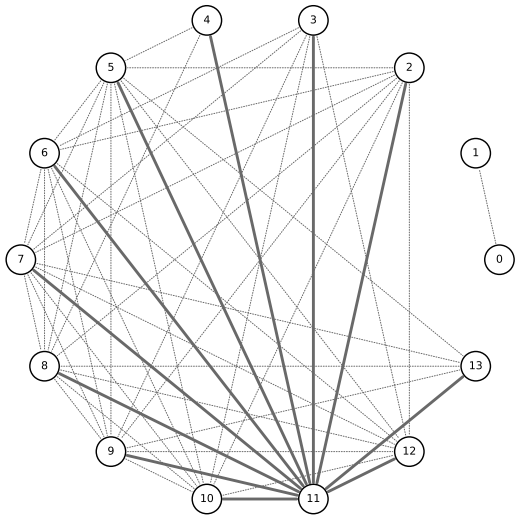

Figure 7: CCZ-switching graph for $n = 6$. The class indices are the ones of the database. For instance, Function 0 is the cube and Function 4 is the Kim mapping. Function 11 is connected to all other functions in its component. The other edges are dotted for clarity.

Role of EA-classes in the Switching-Neighbour-CCZ graph. *In this graph, the CCZ-class of a quadratic function does not reach more CCZ-classes via switching neighbours than the quadratic function itself.* △

Experimental results 7 (Switching neighbours in dimension 8)**.** *We have established the following facts:*

New APN functions. *Before this paper, 3,808,491 quadratic CCZ-inequivalent APN functions were known in dimension 8. Among them, only 28,348 have switching neighbours. Among these neighbours, we found 3,833 new CCZ-classes. All of these functions are quadratic. The supplementary material contains these functions as well as the EA-partition of their CCZ-class.*

Switching-Neighbour-CCZ graph. *The Switching-Neighbour-CCZ graph for the quadratic functions is composed of one large component of size 12,783, and 9,609 other smaller components, whose sizes range between 2 and 20.*

Role of EA-classes in the Switching-Neighbour-CCZ graph. *There are 1.29 billions functions in the EA-partitions of the known APN CCZ-classes in dimension 8. We conducted random trials to assess whether the non-quadratic functions could reach more CCZ-classes than the quadratic functions themselves or not. We drew randomly 1 million functions of degree 3, 1 million functions of degree 4 and all the degree-5 functions in the EA-partitions and computed their switching neighbours. In all cases, these functions reached at most the same CCZ-classes as the quadratic functions in their class.* △

The previous statistical result leads us to the following conjecture.

Conjecture 4. *Let Q be a quadratic APN function and F be any function in its CCZ-class. The set of CCZ-classes that are switching neighbours of F is a subset of the set of CCZ-classes that are switching neighbours of Q.*

Regarding the Switching-Neighbour-CCZ graph, the appearance of such a large component in dimension 6 and 8, and its inexistence in dimension 7 rises the question of the size of the largest component that appears in dimension n. We formalize it as the following open problem.

Problem 1. *What is the largest neighbours-CCZ graph component in dimension n? How does it grow depending on n and its parity?*

We also report some results for dimension 9 and 10.

Experimental results 8 (Switching neighbours in dimension 9)**.** *In dimension 9, the database contains all members of the known infinite families of APN functions from [BCC+20], [BCL09b], [BCL09a],[LZLQ22] and [LK23] as well as the Gold, Kasami and Welsh and Inverse monomials. There are, up to our knowledge, 21 such CCZ-classes. The only non-trivial CCZ-SN component is the one of the cube. It contains 4 functions.* △

Experimental results 9 (Switching neighbours in dimension 10)**.** *In dimension 9, the database contains all members of the known infinite families of APN functions from [BCL09a], [BCV20], [BHK20], [CLV22], [Göl22],[Tan19] and [SPZ25] as well as the Gold, Kasami and Dobbertin monomials. There are, up to our knowledge, 23 such CCZ-classes. There are 3 non-trivial CCZ-SN components: the one of the cube that contains 3 functions and two others, each of them composed of a pair of functions from [SPZ25].*

Remark 14. We want to highlight that [SPZ25] reports 22 quadratic classes distinguished using invariants. We conducted the same computations and found only 21 distinct invariants: pairs 8/18, 9/19, 10/21 and 11/20 collide and the differential spectrum of function 7 is incorrect.

△

5.1.4 Neighbourhood spectrum as an EA invariant

Finally, we quickly mention that the algorithmic and theoretical advances of the previous sections allow us to consider new invariants that perform better than the state-of-the-art mugshots used in the previous experiments. Indeed, as per Corollary 4, the multiset $\{\!\{|\mathsf{sw}_F(u), u \in \mathbb{F}_2^n|\}\!\}$ is an EA-invariant. We call it the *neighbourhood spectrum* of F. The effectiveness of this new invariant is detailed in the following experimental observations.

Experimental results 10 (Neighbourhood spectrum in dimension 6)**.** *We have established the following facts.*

Variety of spectra. *Among the 716 EA-classes, there are 155 distinct neighbourhood spectra.*

Number of functions sharing the same spectrum. *The maximum number of functions having the same spectrum is 39, and the minimum is 1.*

Comparison with previous mugshot. *The previously-mentioned mugshot[5] splits the EA-classes into 160 distinct mugshots. The maximum number of functions sharing a given mugshot is 39. By considering both the neighbourhood spectrum and the previous mugshot, we obtain 261 mugshots and the maximum number of functions sharing the same mugshot is 9.* △

Experimental results 11 (Neighbourhood spectrum in dimension 7)**.** *We established the following facts:*

Variety of spectra. *Among the 82,409 EA-classes in the CCZ-classes of the known quadratic functions, there are 9 distinct spectra.*

Number of functions sharing the same spectrum. *The maximum number of functions having the same spectrum is 72519, and the minimum is 1.*

Comparison with previous mugshot. *The previously-mentioned mugshot splits the 82,409 EA-classes into 4,718 distinct mugshots. The maximum number of functions sharing a mugshot is 1,540. By considering both the neighbourhood spectrum and the previous mugshot, we obtain 5,879 mugshots and the maximum number of functions sharing the same mugshot is 1,233.*

Distinguishing two power mapping. *The function x^3 and x^9 have different neighbourhood spectrum, providing a more efficient way to distinguish them than the method proposed in [ZLZ25].* △

[5] See "Our exploration methodology" paragraph at the beginning of Section 5.

Regarding, the 8-bit case, we did not compute the neighbourhood spectra for the 1.29 billions functions, but instead computed the ones of the quadratic functions. Since quadratic functions are filtered using mugshots, they all have distinct mugshots, so it is useless to add the neighbourhood spectrum to it. Nonetheless, we report the following result.

Experimental results 12 (Neighbourhood spectrum in dimension 8)**.** *We established the following facts:*

Variety of spectra. *Among the 1.29 billions CCZ-classes of known quadratic functions, there are 219 non-trivial distinct spectra.*

Number of functions sharing the same spectrum. *The maximum number of functions having the same spectrum is 19,203, and the minimum is 1.* △

From the previous results, it is clear that the efficiency of the neighbourhood spectrum as an invariant decreases with n. This comes from the fact that the proportion of APN functions F for which SN_F^{nt} is non-empty experimentally decreases with n. As the number of known functions in dimension 9 and 10 is not representative, we do not report the results for these dimensions.

Building CCZ-invariants from EA-partition and EA-invariants. Finally, we would like to mention that, since a CCZ-class splits into multiple EA-classes, it is possible to build CCZ-invariant from the EA-partition.

First the number of distinct EA-classes in a CCZ-class is an invariant, which is now possible to compute from our exact system of representatives. As shown previously, this can be done very efficiently. Looking back at Figures 3, 4 and 6, we can observe that in dimension 6, this provides 8 different size of systems of EA-representatives among 14 classes. The maximum number of functions that share the same size is 3, and the minimum is 1. In dimension 7, there are 51 different size of systems of EA-representatives among 488 classes, with a maximum of 38 functions sharing the same size a minimum of 1.

Remark 15. The efficiency of this invariant is decreasing with n. Since the size of an exact system of EA-representatives is determined by the action of $\mathrm{Aut}(F)$ on the vectors spaces in $\mathcal{Z}_F$ as per Theorem 2, this degrowth seems to be caused by the fact that the number of known APN functions is growing faster than the possible cardinalities of the orbits under this action.

Furthermore, a CCZ-invariant can also be built for a CCZ-class $\mathcal{C}$ by considering the multiset of $\mathrm{Inv}_{\mathrm{CCZ}}(\mathcal{C}) = \{\!\{\mathrm{Inv}_{\mathrm{EA}}(\mathcal{C}') \mid \mathcal{C}' \text{ an EA-class s.t. } \mathcal{C}' \subset \mathcal{C}\}\!\}$, where $\mathrm{Inv}_{\mathrm{EA}}$ is any EA-invariant, *e.g.* a mugshot aggregating simpler EA-invariants. This is again made possible by the fact that an exact system of representatives can now be easily computed. Note that even if an exact partition is not known, a difference of EA-mugshots on a finer partition is enough to distinguish functions. Hence, the computation of the entire automorphisms group might not be essential to the efficiency.

6 Conclusion

In this work we provided new theoretical insights on the way of exploring the set of known APN functions. We improved existing techniques to efficiently explore the distinct EA-classes of a given CCZ-class, but also to explore new CCZ-classes from a known one based on the switching neighbours technique. Both of these theoretical advances comes with their algorithmic counterpart as we implemented every algorithm we described and made them publicly-available as part of the `sboxU` library. This allowed us to build public databases of known APN in small dimensions, to reproduce previously-reported results and to provide many new statistics, as well as all the necessary tools to compute others. More importantly, and despite the few thousands new quadratic APN functions we report in dimension 8, it seems that the combined usage of the EA-partition traversal together with the switching neighbours technique does not scale well when n increases. This is not due to an algorithmic limitation since our implementation still performs well for $n \in \{8, 9, 10\}$, but most often return no new function. Instead, our experimental results seem to indicate that APN functions grow further and further apart as n increases, while these techniques perform "local" searches around a given function.

Acknowledgments. A preliminary version of this paper was presented at the 10th International Workshop on Boolean Functions and their Applications (BFA 2025) [BGP25].

The authors acknowledge that, all the ideas presented in this work are theirs. Some AI tools were exclusively used to generate part of their experimental code. The authors accept responsibility for the veracity and correctness of all material that are presented, including any computer-generated material.

The work of Léo Perrin & Pierre Galissant is supported by the European Research Council (ERC) under grant number 101041545 (ReSCALE).

References

[BBL21] Christof Beierle, Marcus Brinkmann, and Gregor Leander. Linearly self-equivalent APN permutations in small dimension. *IEEE Transactions on Information Theory*, 67(7):4863–4875, 2021.

[BBMM11] Carl Bracken, Eimear Byrne, Nadya Markin, and Gary McGuire. A few more quadratic APN functions. *Cryptography and Communications*, 3(1):43–53, 2011.

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

[BCL08] Lilya Budaghyan, Claude Carlet, and Gregor Leander. Two classes of quadratic APN binomials inequivalent to power functions. *IEEE Transactions on Information Theory*, 54(9):4218–4229, 2008.

[BCL09a] Lilya Budaghyan, Claude Carlet, and Gregor Leander. Constructing new APN functions from known ones. *Finite Fields and Their Applications*, 15(2):150–159, 2009.

[BCL09b] Lilya Budaghyan, Claude Carlet, and Gregor Leander. On a construction of quadratic APN functions. In *2009 IEEE Information Theory Workshop*, pages 374–378, 2009.

[BCV20] Lilya Budaghyan, Marco Calderini, and Irene Villa. On equivalence between known families of quadratic APN functions. *Finite Fields and Their Applications*, 66:101704, 2020.

[BD94] Thomas Beth and Cunsheng Ding. On almost perfect nonlinear permutations. In Tor Helleseth, editor, *EUROCRYPT'93*, volume 765 of *LNCS*, pages 65–76. Springer, Berlin, Heidelberg, May 1994.

[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.

[BGP25] Jules Baudrin, Pierre Galissant, and Léo Perrin. Exploring the Set of APN Functions. In *BFA 2025 - 10th International Workshop on Boolean Functions and their Applications*, Larnaca, Cyprus, September 2025.

[BHK20] Lilya Budaghyan, Tor Helleseth, and Nikolay Kaleyski. A new family of APN quadrinomials. *IEEE Transactions on Information Theory*, 66(11):7081–7087, 2020.

[BL08] Marcus Brinkmann and Gregor Leander. On the classification of APN functions up to dimension five. *Designs, Codes and Cryptography*, 49(1):273–288, Dec 2008.

[BL22] Christof Beierle and Gregor Leander. New instances of quadratic APN functions. *IEEE Transactions on Information Theory*, 68(1):670–678, 2022.

[BLL+25] Christof Beierle, Philippe Langevin, Gregor Leander, Alexandr Polujan, and Shahram Rasoolzadeh. Millions of inequivalent quadratic APN functions in eight variables. *CoRR*, abs/2508.04644, 2025.

[BPT19] Xavier Bonnetain, Léo Perrin, and Shizhu Tian. Anomalies and vector space search: Tools for S-box analysis. In *Advances in Cryptology – ASIACRYPT 2019*, Cham, 2019. Springer International Publishing.

[BS91] Eli Biham and Adi Shamir. Differential cryptanalysis of Feal and N-hash. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 1–16. Springer, Berlin, Heidelberg, April 1991.

[BS93] Eli Biham and Adi Shamir. Differential cryptanalysis of the full 16-round DES. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 487–496. Springer, Berlin, Heidelberg, August 1993.

[Cal20] Marco Calderini. On the EA-classes of known APN functions in small dimensions. *Cryptogr. Commun.*, 12(5):821–840, 2020.

[Car21] Claude Carlet. *Boolean Functions for Cryptography and Coding Theory*. Cambridge University Press, 2021.

[CCP22] Anne Canteaut, Alain Couvreur, and Léo Perrin. Recovering or testing extended-affine equivalence. *IEEE Transactions on Information Theory*, 68(9):6187–6206, 2022.

[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.

[CL22] Benjamin Chase and Petr Lisonek. Construction of APN permutations via walsh zero spaces. *Cryptogr. Commun.*, 14(6):1345–1357, 2022.

[CLV22] Marco Calderini, Kangquan Li, and Irene Villa. Two new families of bivariate APN functions. *CoRR*, abs/2204.07462, 2022.

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

[DE14] Ulrich Dempwolff and Yves Edel. Dimensional dual hyperovals and APN functions with translation groups. *Journal of Algebraic Combinatorics*, 39:457–496, 2014.

[Dob99a] Hans Dobbertin. Almost perfect nonlinear power functions on GF(2^n): The Niho case. *Inf. Comput.*, 151(1-2):57–72, 1999.

[Dob99b] Hans Dobbertin. Almost perfect nonlinear power functions on GF(2^n): The Welch case. *IEEE Transactions on Information Theory*, 45(4):1271–1275, 1999.

[Dob01] Hans Dobbertin. Almost perfect nonlinear power functions on GF(2^n): A new case for n divisible by 5. In Dieter Jungnickel and Harald Niederreiter, editors, *Finite Fields and Applications*, pages 113–121, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[EP09] Yves Edel and Alexander Pott. A new almost perfect nonlinear function which is not quadratic. *Adv. Math. Commun.*, 3(1):59–81, 2009.

[GK25] Faruk Göloglu and Lukas Kölsch. Equivalences of biprojective almost perfect nonlinear functions. *Comb. Theory*, 5(3), 2025.

[GL26a] Valérie Gillot and Philippe Langevin. On Known APNs. *CoRR*, abs/2601.11247, 2026.

[GL26b] Valérie Gillot and Philippe Langevin. New Invariants for Boolean and Vectorial Functions. In *BFA 2026: The 11th International Workshop on Boolean Functions and their Applications*, Sogndal, Norway, September 2026. Selmer Center. https://boolean.w.uib.no/bfa-2026/.

[GLL25] Valérie Gillot, Philippe Langevin, and Addoulaye Lo. on the Dublin permutation. In *BFA 2025: The 10th International Workshop on Boolean Functions and their Applications*, Lacarna, Cyprus, September 2025. Selmer Center. https://boolean.w.uib.no/bfa-2025/.

[Göl22] Faruk Göloğlu. Biprojective almost perfect nonlinear functions. *IEEE Transactions on Information Theory*, 68(7):4750–4760, 2022.

[Göl23] Faruk Göloğlu. Classification of (q, q)-biprojective APN functions. *IEEE Transactions on Information Theory*, 69(3):1988–1999, 2023.

[HK22] Marie Heggebakk and Nikolay Kaleyski. Testing linear and affine equivalence. In *BFA 2022: The 7th International Workshop on Boolean Functions and their Applications*, Balestrand, Norway, September 2022. https://boolean.w.uib.no/bfa-2022/.

[JKK25] Jaeseong Jeong, Namhun Koo, and Soonhak Kwon. On the functions which are CCZ-equivalent but not EA-equivalent to quadratic functions over $\mathbb{F}_{p^n}$. *Finite Fields and Their Applications*, 103:102574, 2025.

[KI23] Konstantin Kalgin and Valeriya Idrisova. The classification of quadratic APN functions in 7 variables and combinatorial approaches to search for APN functions. *Cryptogr. Commun.*, 15(2):239–256, 2023.

[KS26] Nikolay Kaleyski and Joakim Sunde. Efficiently deciding and recovering CCZ and EA equivalence for arbitrary vectorial boolean functions using the partition refinement framework. Cryptology ePrint Archive, Paper 2026/940, 2026.

[KZ21] Christian Kaspers and Yue Zhou. The number of almost perfect nonlinear functions grows exponentially. *J. Cryptol.*, 34(1):4, 2021.

[KZ22] Christian Kaspers and Yue Zhou. A lower bound on the number of inequivalent APN functions. *J. Comb. Theory A*, 186:105554, 2022.

[LK23] Kangquan Li and Nikolay Kaleyski. Two new infinite families of APN functions in trivariate form. *IEEE Transactions on Information Theory*, 2023.

[LSS12] Philippe Langevin, Elif Saygi, and Zulfukar Saygi. Classification of APN cubics in dimension 6 over GF(2), February 2012. https://langevin.univ-tln.fr/project/apn-6/apn-6.html.

[LZLQ22] Kangquan Li, Yue Zhou, Chunlei Li, and Longjiang Qu. Two new families of quadratic APN functions. *IEEE Transactions on Information Theory*, 68(7):4761–4769, 2022.

[Nyb91] Kaisa Nyberg. Constructions of Bent functions and difference sets. In Ivan Damgård, editor, *EUROCRYPT'90*, volume 473 of *LNCS*, pages 151–160. Springer, Berlin, Heidelberg, May 1991.

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

[PAB+26] Léo Perrin, Jens Alich, Jules Baudrin, Aurélien Boeuf, Xavier Bonnetain, Alain Couvreur, Merlin Fruchon, Pierre Galissant, Guilhem Jazeron, Mathias Joly, and Lukas Stennes. sboxU: black-box analysis of discrete functions. Available online at https://github.com/lpp-crypto/sboxU/, 2026.

[SPZ25] Chenmiao Shi, Jie Peng, and Lijing Zheng. A new infinite family of bivariate APN multinomials. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 108(9):1327–1330, 2025.

[Tan19] Hiroaki Taniguchi. On some quadratic APN functions. *Designs, Codes and Cryptography*, 87:1973–1983, 2019.

[Yos12] Satoshi Yoshiara. Equivalences of quadratic apn functions. *Journal of Algebraic Combinatorics*, 35:461–475, 2012.

[YP22] Yuyin Yu and Léo Perrin. Constructing more quadratic APN functions with the QAM method. *Cryptography and Communications*, 14(6):1359–1369, 2022.

[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.

[ZKL+22] Lijing Zheng, Haibin Kan, Yanjun Li, Jie Peng, and Deng Tang. Constructing new APN functions through relative trace functions. *IEEE Transactions on Information Theory*, 68(11):7528–7537, 2022.

[ZLZ25] Zijian Zhou, Kangquan Li, and Yue Zhou. Topological invariants for linear codes and APN functions. *IEEE Trans. Inf. Theory*, 71(9):6771–6784, 2025.

[ZP13] Yue Zhou and Alexander Pott. A new family of semifields with 2 parameters. *Advances in Mathematics*, 234:43–60, 2013.