Noname manuscript No.
(will be inserted by the editor)

Construction of quadratic APN functions with coefficients in $\mathbb{F}_2$ in dimensions 10 and 11

Yuyin Yu[1] · **Jingchen Li**[1] · **Nadiia Ichanska**[2] · **Nikolay Kaleyski**[2]

Received: date / Accepted: date

Abstract Yu et al. described an algorithm for conducting computational searches for quadratic APN functions over the finite field $\mathbb{F}_{2^n}$, and used this algorithm to give a classification of all quadratic APN functions with coefficients in $\mathbb{F}_2$ for dimensions n up to 9. In this paper, we speed up the running time of that algorithm by a factor of approximately $\frac{n\times 2^n}{n^3}$. Based on this result, we give a complete classification of all quadratic APN functions over $\mathbb{F}_{2^{10}}$ with coefficients in $\mathbb{F}_2$. We also perform some partial computations for quadratic APN functions over $\mathbb{F}_{2^{11}}$ with coefficients in $\mathbb{F}_2$, and conjecture that they form 6 CCZ-inequivalent classes which also correspond to known APN functions.

Keywords Boolean functions, Almost Perfect Nonlinear, ortho-derivative, Quadratic functions

1 Introduction

Let n be a positive integer. We denote by $\mathbb{F}_{2^n}$ the finite field with 2^n elements, and by $\mathbb{F}_{2^n}^\star$ its multiplicative group; we will refer to n as the **dimension** of the finite field. A mapping $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ is called an (n, n)**-function**. An (n, n)-function F is said to have **differential uniformity** $\delta(F)$ if

$$\delta(F) = \max_{a\in\mathbb{F}_{2^n}^\star, b\in\mathbb{F}_{2^n}} \#\Delta_F(a, b),$$

where $\Delta_F(a, b) = \{x \in \mathbb{F}_{2^n} : F(x + a) + F(x) = b\}$, and $\#S$ is the cardinality of a set S. If $\delta(F) \leq d$ for some positive integer d, then we say that F is **differentially** d**-uniform**. From the point of view of cryptography, where (n, n)-functions are frequently used as so-called *substitution boxes*, or *S-boxes*,

1. College of Mathematics and Information Science, Guangzhou University, Guangzhou (yuyuyin@163.com; lijingchen0702@qq.com)
2. Department of Informatics, University of Bergen (Nadiia.Ichanska@uib.no; Nikolay.Kaleyski@uib.no)

in the design of symmetric block ciphers, it is desirable for $\delta(F)$ to be as low as possible since this indicates a better resistance to differential cryptanalysis. It is easy to see that $\delta(F)$ is always even and can not be equal to zero, and therefore the lowest value of $\delta(F)$ is 2. If $\delta(F) = 2$, then F is called **almost perfect nonlinear (APN)**. Consequently, APN functions provide the best possible resistance to differential attacks, and have attracted significant attention from researchers for this reason. However, as a "side effect" of their cryptographic strength, APN functions tend to have little obvious structure or properties that can be used to construct or characterize them. Finding APN functions is thus not an easy task, and it becomes even more challenging to find functions that combine APN-ness with other desirable properties. Ideally, one would have a classification of all APN functions for a given dimension n, but this is an extremely challenging task, even if only functions from some particular subclass are considered. The vast majority of known APN functions are quadratic, and most of the literature on APN functions concerns primarily quadratic APN functions. Considering quadratic functions all of whose coefficients belong to $\mathbb{F}_2$ further simplifies the classification problem, although it does remain very hard. The last results in this direction are from 2020 [26] where a classification of quadratic APN functions over $\mathbb{F}_{2^n}$ with coefficients in $\mathbb{F}_2$ is performed for $n \leq 9$, and it is shown that applying the approach described in that work to dimensions beyond 9 is infeasible. In this paper, we improve the method from [26], which significantly speeds up the computation and allows us to obtain a complete classification for $n = 10$ and some partial results for $n = 11$; based on these, we also conjecture the exact form of the classification for $n = 11$.

We now give a brief survey of previous classifications and searches for APN functions. Edel et al. [17] found over $\mathbb{F}_{2^{10}}$ the first quadratic APN function that is CCZ-inequivalent [15] to any power function. Brinkmann and Leander [5] showed that when $n < 6$, all APN functions on $\mathbb{F}_{2^n}$ are CCZ-equivalent to power functions, and the APN functions listed in [18] already contain all APN power functions for $n = 4$ and $n = 5$. Dillon et al. [6] first listed 13 CCZ-inequivalent quadratic APN functions on $\mathbb{F}_{2^6}$, while Edel [16] proved that Dillon's list in fact includes all CCZ-inequivalent classes of quadratic APN functions over $\mathbb{F}_{2^6}$.

When $n = 7$, Dillon et al. [6] listed 15 CCZ-inequivalent quadratic APN functions on $\mathbb{F}_{2^7}$, while Edel and Pott [18] expanded this number to 16. Yu et al. [28] found another 471 new CCZ-inequivalent classes. Kalgin and Idrisova [19] found another new APN function and showed that all quadratic APN functions on $\mathbb{F}_{2^7}$ are included in these 488 CCZ-inequivalent classes.

When $n = 8$, Dillon et al. [6] listed 11 CCZ-inequivalent quadratic APN functions on $\mathbb{F}_{2^8}$, while Edel and Pott [18] expanded this number to 22. Yu et al. [27][28] found 8157 new CCZ-inequivalent classes by using a computational search based on a matrix representation of quadratic functions[1]. Weng et al.

[1] The *eprint* version [27] was updated with additional computational results after the publication of the journal version [28], hence why we cite both versions

[23] found 10 new ones. Taniguchi [21] found 2 new ones, and Budaghyan et al. [7] found yet another new one. Further, Beierle et al. [1][2] found 12921 new CCZ-inequivalent APN functions using a computational approach based on so-called self-equivalences. Yu and Perrin [25] found another 5412 new APN functions and conjectured that the total numer would exceed 50000. Consequently, Beierle et al. [4] found 6368 new APN functions. Up to now, there are 32893 known classes of CCZ-inequivalent quadratic APN functions over $\mathbb{F}_{2^8}$.

Yu et al. [26] described an improvement of the algorithm from [28] in the case when the functions have coefficients in $\mathbb{F}_2$, and gave a classification of all quadratic APN functions with coefficients in $\mathbb{F}_2$ for dimensions up to 9. Beierle et al. [1][2] found 35 CCZ-inequivalent quadratic APN functions on $\mathbb{F}_{2^9}$ using the self-equivalence approach.

By the year 2022, a total of 20 CCZ-inequivalent quadratic APN instances over $\mathbb{F}_{2^{10}}$ have been identified. Besides the functions equivalent to monomials, 15 quadratic CCZ-inequivalent APN instances over $\mathbb{F}_{2^{10}}$ given by [10], [11], [12], [20], and [21]. Recent progress has been achieved in [2], in which 5 new APN instances were found in dimension 10 (the look-up tables of these 5 APN instances are available in [3]). Shortly thereafter, Lijing Z. et al. [29] reported 4 CCZ- inequivalent APN instances over $\mathbb{F}_{2^{10}}$. Budaghyan et al. [8] obtained 6 previously unknown classes of APN functions by adding 5, 6, or 7 terms with coefficients in $\mathbb{F}_{2^2}$ to x^3, x^9, or x^{33}. They verified that none of these functions are CCZ-equivalent to a permutation by using SboxU [22].

Many scholars have also studied how to construct infinite families of quadratic APN polynomials over $\mathbb{F}_{2^n}$. Readers can refer to [14] for a comprehensive summary on the state of knowledge of APN functions.

In this paper, we concentrate on quadratic homogeneous functions

$$F(x) = \sum_{0 \le i < j \le n-1} c_{i,j} x^{2^i + 2^j},$$

over $\mathbb{F}_{2^n}$ where $c_{i,j} \in \mathbb{F}_2$, and we investigate when they are APN. We speed up the algorithm in [26] by a factor of approximately $\frac{n \times 2^n}{n^3}$ times. Based on this result, we were able to conduct an exhaustive search in dimension 10. After running 22 Magma [30] scripts in parallel for about 100 days on $\mathbb{F}_{2^{10}}$, we have obtained 29088 quadratic homogeneous APN functions with coefficients in $\mathbb{F}_2$. Using the differential spectra of their ortho-derivatives [13], these APN functions can be divided into 3 CCZ-inequivalent classes corresponding to the functions x^3, x^9 and $x^3 + \mathrm{Tr}(x^9)$, where the trace function $\mathrm{Tr} : \mathbb{F}_{2^n} \to \mathbb{F}_2$ is defined by $\mathrm{Tr}(x) = \sum_{i=0}^{n-1} x^{2^i}$ for $x \in \mathbb{F}_{2^n}$. Using the equivalence algorithm from [9], we show that any two of these functions having the same differential spectrum of the ortho-derivative are CCZ-equivalent to each other. Thus, all quadratic APN functions over $\mathbb{F}_{2^{10}}$ with coefficients in $\mathbb{F}_2$ fall into these three CCZ-equivalence classes. However, we found a shorter representative $x^{513} + x^{192} + x^{96} + x^{48} + x^{33} + x^{18} + x^{12}$ for $x^3 + \mathrm{Tr}(x^9)$. This relates to Problem 1 proposed in [26], namely: given an (n, n)-function F, find a function G CCZ-equivalent to it having the shortest possible number of terms in its polynomial

representation. Based on this, we can confirm that any quadratic APN function over $\mathbb{F}_{2^n}$ with coefficients in $\mathbb{F}_2$ is CCZ-equivalent to a function having no more than n terms with non-zero coefficients for all dimensions n up to 10.

When $n = 11$, a complete classification does not appear to be feasible, but we perform a partial search and find 6 CCZ-inequivalent quadratic APN functions with coefficients in $\mathbb{F}_2$. These functions are all CCZ-equivalent to known ones. We conjecture that these are the only 6 CCZ-inequivalent quadratic APN functions over $\mathbb{F}_{2^{11}}$ with coefficients in $\mathbb{F}_2$.

2 Preliminaries

We introduce some notions and results which will be useful in the sequel.

We say that an (n, n)-function A is **affine** if for any $x, y, z \in \mathbb{F}_{2^n}$ we have $A(x) + A(y) + A(z) = A(x + y + z)$. If an affine function satisfies $A(0) = 0$, then we say that it is **linear**.

Definition 1 Let F and F' be two functions from $\mathbb{F}_{2^n}$ to $\mathbb{F}_{2^n}$. We say that F and F' are **EA-equivalent** (Extended affine equivalent) if we can write F' as

$$F'(x) = A_1(F(A_2(x))) + A_3(x),$$

where A_1 and A_2 are affine permutations of $\mathbb{F}_{2^n}$, and A_3 is an affine function over $\mathbb{F}_{2^n}$.

We say that F and F' are **CCZ-equivalent** (Carlet-Charpin-Zinoviev equivalent) [15], if there exists an affine permutation of $\mathbb{F}_{2^n} \times \mathbb{F}_{2^n}$, or equivalently $\mathbb{F}_{2^{2n}}$, which maps G_F onto $G_{F'}$, where $G_F = \{(x, F(x)) : x \in \mathbb{F}_{2^n}\}$ is the graph of F.

Definition 2 We denote by $\mathcal{F}_n$ a maximal subset of $\mathbb{F}_{2^n}^*$ such that $x \in \mathcal{F}_n$ implies $x^{2^k} \notin \mathcal{F}_n$ for any $1 \leq k \leq n - 1$.

Later in the paper, we will introduce an equivalence relation which we call "cyclic equivalence" between the elements of $\mathbb{F}_{2^n}^*$. A set $\mathcal{F}_n$ will then be a set of representatives from the equivalence classes induced by this relation on $\mathbb{F}_{2^n}^*$.

Definition 3 [28] Let $B \in \mathbb{F}_{2^n}^m$ be a vector $B = (\eta_0, \eta_1, \ldots, \eta_{m-1})$ where $\eta_i \in \mathbb{F}_{2^n}$ for $0 \leq i \leq m - 1$. Then $\mathrm{Span}(B) = \mathrm{Span}(\eta_0, \eta_1, \ldots, \eta_{m-1})$ is the subspace spanned by $\{\eta_0, \eta_1, \ldots, \eta_{m-1}\}$ over $\mathbb{F}_2$. The dimension of this subspace is denoted by $\mathrm{Rank}(B) = \mathrm{Rank}(\eta_0, \eta_1, \ldots, \eta_{m-1})$, and is referred to as the **rank** of B over $\mathbb{F}_2$.

Suppose $\{\alpha_0, \alpha_1, \ldots, \alpha_{n-1}\}$ is a basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$, and $\eta_i = \sum_{j=0}^{n-1} \lambda_{i,j}\alpha_j$ for $0 \leq i \leq m - 1$, with $\lambda_{i,j} \in \mathbb{F}_2$ for $0 \leq i, j \leq n - 1$. We define an m-by-n matrix $\Lambda \in \mathbb{F}_2^{m \times n}$ by $\Lambda[i, j] = \lambda_{i,j}$, so that each row of Λ contains the coordinates of η_i with respect to the basis. Then the rank of B is equal to the rank of Λ.

While the above is true for any basis, we will assume throughout the rest of the paper that $\{\alpha_0, \alpha_1, \ldots, \alpha_{n-1}\}$ is a normal basis. This allows us to exploit the diagonal properties of QAMs described below, and simplifies some computations.

Definition 4 [28] Let $H \in \mathbb{F}_{2^n}^{m \times k}$ with $1 \leq m, k \leq n$. We say that H is **proper** if every nonzero $\mathbb{F}_2$-linear combination of the m rows of H has rank at least $k-1$.

Definition 5 [28] Let H be an $n \times n$ matrix defined on $\mathbb{F}_{2^n}$. Then H is called a **QAM** (quadratic APN matrix) if:
i) H is symmetric and the elements in its main diagonal are all zeros;
ii) H is proper, i.e. every nonzero linear combination of the n rows (or, equivalently, columns, due to H being symmetric) of H has rank $n-1$.

Theorem 1 *[26] Let* $F(x) = \sum_{0 \leq t < i \leq n-1} c_{i,t} x^{2^i + 2^t}$ *be a quadratic homogeneous* (n,n)*-function. Define an* $n \times n$ *matrix* C_F *by* $C_F[t,i] = C_F[i,t] = c_{i,t}$ *for* $0 \leq t < i \leq n-1$ *and* $C_F[i,i] = 0$ *for* $0 \leq i \leq n-1$*. Finally, take*

$$H = M_\alpha^t C_F M_\alpha.$$

Then

$$H[u+1, v+1] = H[u,v]^2 \tag{1}$$

(with the indices taken modulo n*) for* $0 \leq u, v \leq n-1$ *if and only if* $c_{i,t} \in \mathbb{F}_2$ *for* $0 \leq t < i \leq n-1$*.*

Intuitively, when the coefficients of the function all belong to $\mathbb{F}_2$, the values in H have the property that, as we go down any diagonal, each subsequent value on that diagonal is simply the square of the previous value. Thus, knowing a single value of H gives us knowledge of the values on its entire diagonal.

Proposition 1 *[26] Suppose* $F_1 \in \mathbb{F}_{2^n}[x]$ *is a homogeneous quadratic APN function with coefficients in* $\mathbb{F}_2$*, and* H *is its corresponding QAM. Let* H' *be the matrix defined by* $H'[i,j] = H[i,j]^2$ *for* $0 \leq i, j < n$*. Then* H' *is also a QAM, and its corresponding function* $F_2 \in \mathbb{F}_2[x]$ *is EA-equivalent to* F_1*.*

According to [24], two quadratic APN functions are CCZ-equivalent if and only if they are EA-equivalent. In order to categorize our functions into distinct EA-equivalence classes (and thus distinct CCZ-classes), we computed the values of the differential spectra of the ortho-derivatives [13] for each function, and then applied the algorithm from [9] to every pair of functions having the same ortho-derivative differential spectrum. First, let us recall the definition of the ortho-derivative.

Definition 6 [13] Let $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ be a quadratic APN function, and let $x \cdot y$ denote a scalar product, that is, a symmetric bivariate function on $\mathbb{F}_{2^n}$

such that $x \mapsto a \cdot x$ is a non-zero linear form for any $a \in \mathbb{F}_{2^n}^*$. Then the *ortho-derivative* of F is the unique function $\pi_F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ such that $\pi_F(0) = 0$, $\pi_F(a) \neq 0$ if $a \neq 0$, and such that

$$\pi_F(a) \cdot \big(F(x+a) + F(x) + F(a) + F(0)\big) = 0 \tag{2}$$

for all $a \in \mathbb{F}_{2^n}^*$ and all $x \in \mathbb{F}_{2^n}$.

According to [13], if two quadratic APN functions are EA-equivalent, then their ortho-derivatives are affine equivalent, which implies that their ortho-derivatives must have the same extended Walsh spectra and differential spectra. Thus, if the ortho-derivatives of two quadratic APN functions have different extended Walsh spectra or differential spectra, then they are EA-inequivalent. It is known that the Walsh and differential spectra of the ortho-derivatives are very efficient for distinguishing between CCZ-inequivalent quadratic APN functions. Since they can be computed very quickly, doing so as a first step before applying an equivalence algorithm can significantly speed up the process of categorizing a set of functions into CCZ-equivalence classes.

3 A faster method to determine QAM

According to the definition of QAM (See Definition 5), if an $n \times n$ matrix H is QAM, then every nonzero linear combination of the n rows of H has rank $n-1$. Note that there are $2^n - 1$ nonzero linear combinations of the n rows of H. In this section, we will investigate how to reduce the data to be tested for determining quadratic APN functions with coefficients in $\mathbb{F}_2$.

3.1 Reducing the data to be tested

We will prove that about $\frac{2^n}{n}$ nonzero linear combinations of the n rows of H need to be tested for determining quadratic APN functions with coefficients in $\mathbb{F}_2$. Intuitively, we already know that "going down diagonals" in H results in all values getting squared; and so, the values in two adjacent rows are simply squares of each other, implying that any two adjacent rows have the same rank.

Theorem 2 *Let $H \in \mathbb{F}_{2^n}^{n \times n}$, and $H[u,u] = 0$, $H[u,v] = H[v,u]$, $H[u+1, v+1] = H[u,v]^2$ (with the indices taken modulo n) for $0 \leq v, u \leq n-1$. Then*

$$Rank(\sum_{u=0}^{n-1} \lambda_u H[u,:]) = Rank(\sum_{u=0}^{n-2} \lambda_u H[u+1,:] + \lambda_{n-1} H[0,:]), \tag{3}$$

where $(\lambda_0, \lambda_1, \ldots, \lambda_{n-1}) \in \mathbb{F}_2^n \backslash \{0\}$, and $H[u,:]$ is the $u-th$ row of H.

Proof According to the properties of H, we have

$$Span(\sum_{u=0}^{n-2} \lambda_u H[u+1,:] + \lambda_{n-1} H[0,:]) = Span((\sum_{u=0}^{n-1} \lambda_u H[u,:])^2)$$

which implies that

$$\begin{aligned} Rank(\sum_{u=0}^{n-2} \lambda_u H[u+1,:] + \lambda_{n-1} H[0,:]) &= Rank((\textstyle\sum_{u=0}^{n-1} \lambda_u H[u,:])^2 \\ &= Rank((\textstyle\sum_{u=0}^{n-1} \lambda_u H[u,:]) \end{aligned}$$

Example 1 Let $n = 5$, define

$$H = \begin{pmatrix} 0 & a & b & b^8 & a^{16} \\ a & 0 & a^2 & b^2 & b^{16} \\ b & a^2 & 0 & a^4 & b^4 \\ b^8 & b^2 & a^4 & 0 & a^8 \\ a^{16} & b^{16} & b^4 & a^8 & 0 \end{pmatrix} \in \mathbb{F}_{2^5}^{5\times 5}.$$

(i) Let $(\lambda_0, \lambda_1, \ldots, \lambda_{n-1}) = (1, 0, 0, 0, 0)$. Then

$$H[0,:]^2 = (0, a^2, b^2, b^{16}, a) \text{ and } H[1,:] = (a, 0, a^2, b^2, b^{16}),$$

which implies that

$$Span(H[0,:]^2) = Span(H[1,:]).$$

Therefore, we have

$$Rank(H[1,:]) = Rank(H[0,:]^2) = Rank(H[0,:]).$$

Similarly, we have

$$Rank(H[1,:]) = Rank(H[2,:]) = Rank(H[3,:]) = Rank(H[4,:]).$$

(ii) Let $(\lambda_0, \lambda_1, \ldots, \lambda_{n-1}) = (1, 1, 0, 0, 0)$, Similiarly as in (i), we have

$$Span((H[0,:] + H[1,:])^2) = Span(H[1,:] + H[2,:]).$$

Therefore,

$$\begin{aligned} Rank(H[0,:] + H[1,:]) &= Rank(H[1,:] + H[2,:]) \\ &= Rank(H[2,:] + H[3,:]) \\ &= Rank(H[3,:] + H[4,:]) \\ &= Rank(H[4,:] + H[1,:]). \end{aligned}$$

Based on Theorem 2 and Example 1, we can conclude that only about $\frac{2^n}{n}$ nonzero linear combinations need to be tested. We will give an exact number in the following.

Two elements $a = (a_0, a_1, \ldots, a_{n-1}), b = (b_0, b_1, \ldots, b_{n-1}) \in \mathbb{F}_2^n$ are called **cyclically equivalent** if $a = (b_k, b_{k+1}, \ldots, b_{k+n-1})$ (with the indices taken modulo n) for some k. We will refer to this equivalence relation as **cyclic equivalence**. Let S_t for $t = 0, 1, \ldots, K$ for some positive integer K be all equivalence classes of $\mathbb{F}_{2^n}$ under cyclic equivalence, so that $S_0, S_1, \ldots, S_K$ is the partition (quotient set) of $\mathbb{F}_{2^n}$ induced by cyclic equivalence. According to Theorem 2, we can get the following corollary.

Corollary 1 *Let $S_0 = \{0\}$, and $S_0, S_1, \ldots, S_K$ be the quotient set induced by cyclic equivalence. Suppose H is the same as in Theorem 2, then K nonzero linear combinations of the n rows of H are enough to determine whether it is QAM.*

Proof In fact, only one element from each S_t $(1 \leq t \leq K)$ needs to be considered in order to test the APN property of H. According to Theorem 2, for any cyclically equivalent elements $a = (a_0, a_1, \ldots, a_{n-1}), b = (b_0, b_1, \ldots, b_{n-1}) \in S_i$, we must have

$$\mathrm{Rank}(\sum_{u=0}^{n-1} a_u H[u,:]) = \mathrm{Rank}(\sum_{u=0}^{n-1} b_u H[u,:]).$$

Note that $K \approx \frac{2^n}{n}$. Thus, we can reduce the computation time by a factor of about $\frac{n-1}{n}$, however, the exact speed up heavily depends on how we implement the algorithm. For example, suppose we must compute $\mathrm{Rank}(H[0,:] + H[1,:])$, $\mathrm{Rank}(H[0,:] + H[1,:] + H[2,:])$ and $\mathrm{Rank}(H[0,:] + H[1,:] + H[2,:] + H[3,:])$, then $H[0,:] + H[1,:]$ would be calculated 3 times. Avoiding such redundant computations is a key factor in speeding up the computation.

3.2 Gaussian elimination

The most time-consuming part of the algorithm in [26] is testing the QAM property of a given matrix. According to Definition 3 and Definition 5, we have two ways to speed up that algorithm, one is to reduce the data that needs to be tested, which is what we have done in Section 3.1; the other is to speed up the computation of rank, which is what we want to do in the following.

Based on the definition of rank (See Definition 3), if $B = (\eta_0, \eta_1, \ldots \eta_{m-1}) \in \mathbb{F}_{2^n}^m$, then $\mathrm{Rank}(B)$ is the dimension of $\mathrm{Span}(B)$. Therefore, the time complexity of computing $\mathrm{Rank}(B)$ is $O(2^n)$ by generating $\mathrm{Span}(B)$. However, if we expand B as a $m \times n$ matrix $\Lambda \in \mathbb{F}_2^{m \times n}$, and then calculate the rank of Λ, the time complexity is $O(nm^2)$. This is the second improvement in our paper.

3.3 Algorithm to determine QAM

Considering Sections 3.1 and 3.2 we give a faster algorithm to enumerate all QAMs. The core of the procedure is the function IsQAM(), which is given in pseudocode in Algorithm 1 along with some auxiliary functions necessary for its implementation. The implementation incorporates the ideas for speeding up the computation described in Sections 3.1 and 3.2.

The IsQAM(H) function takes a matrix H and decides whether it is a QAM by computing the rank of linear combinations of its rows. As discussed in Section 3.1, only certain linear combinations need to be considered, which reduces the computation time. The linear combinations are stored in the array RC (short for "row combinations"). This allows us to avoid redundant additions by reusing linear combinations that have already been computed; for example, if one linear combination that we need to test is $H[1,:] + H[2,:]$ and another is $H[1,:] + H[2,:] + H[4,:]$, then the latter can be obtained from the former (which is stored in RC) by simply adding the fourth row of H to it, without having to recompute the sum $H[1,:] + H[2,:]$.

We need to determine which rows of H need to be summed with which entries of RC in order to obtain all linear combinations that need to be checked. In order to do this, we first need to compute a list of representatives $\mathcal{F}_n$ of the elements of $\mathbb{F}_{2^n}^*$ with respect to cyclic equivalence. This is done using the getCyclicEquivalenceRepresentatives() function and is stored in the array CR (short for "cyclic representatives") so that the representatives are in ascending order. We note that every element $(a_0, a_1, \ldots, a_{n-1})$ can be identified with the integer $\sum_{i=0}^{n-1} a_i \times 2^i \in \mathbb{Z}_{2^n}$. The sets S_i forming the quotient set of $\mathbb{F}_{2^n}^*$ with respect to cyclic equivalence can thus be seen as sets of integers, and in the algorithm we select the smallest integer within each set as its representative.

The binary expansion of the representatives precisely corresponds to the linear combinations of rows of H that need to be tested. If $\sum_{i=0}^{n-1} a_i \times 2^i$ is one such representative, it corresponds to the linear combination $\sum_{i=0}^{n-1} H[i,:]$. Thus, computing $\mathcal{F}_n$ is useful both for reducing the number of guesses for the first unknown in H (that is, the first level of the depth-first search), as well as for implementing the verification of the necessary conditions for H to be a QAM.

In order to avoid redundant additions when computing the linear combinations of the rows, we compute an array of indices $ID[i,j]$ for $i \in \{1,2\}$, $j \in \{1,2,\ldots,K\}$. These indices describe how the linear combinations of the rows of H should be iteratively computed in order to avoid redundant computations. The linear combinations themselves are stored in the RC array, which is initialised at the beginning of the computation as containing only the zero vector, i.e. $RC[1]$ is the zero vector. Subsequent linear combinations are computed as $RC[t+1] = RC[ID[1,t]+1] + H[ID[2,t],:]$ for $t = 1,2,\ldots,K$.

To calculate the index array ID, we observe that if $CR[t] = \sum_{i=1}^{n} a_i \times 2^{i-1}$, then $RC[t+1] = \sum_{i=1}^{n} a_i H[i,:]$. When constructing ID, we thus ensure that $CR[t] = CR[ID[1,t]] + 2^{ID[2,t]-1}$, which implies $RC[t+1] = RC[ID[1,t]+1] + H[ID[2,t],:]$.

For every linear combination of the rows, we need to check whether it has rank $n-1$. To do so, after calculating the value of $RC[t+1]$, we expand it to an $n \times n$ matrix over $\mathbb{F}_2$, then transform it to row echelon form by Gaussian elimination method, from which we can get the rank of $RC[t+1]$.

In summary, we first select the smallest representative element in each cyclic equivalence set, and then establish an iterative relationship between these representative elements. This iterative relationship satisfies the following two properties: first, it allows us to test only one representative element in each cyclic equivalence set, which reduces the amount of data that needs to be tested; second, when calculating the sum of any $k+1$ rows of H, only one row addition needs to be done, which avoids redundant row additions.

For clarity, we give an example to show what happens when running IsQAM(n,H) in Algorithm 1.

Example 2 Let $n = 5$, and H be the same as in Example 1. We have $RC[1] = (0,0,0,0,0)$, $CR = getCyclicEquivalenceRepresentatives(n) = (1,3,5,7,11,15,31)$, $IDlen = 7$, and

$$ID = GetIndices(n) = \begin{pmatrix} 0 & 1 & 1 & 2 & 2 & 4 & 6 \\ 1 & 2 & 3 & 3 & 4 & 4 & 5 \end{pmatrix}.$$

Firstly, RC is used to store the nonzero linear combinations of the 5 rows of H which need to be checked. Based on $RC[t+1] = RC[ID[1,t]+1]+H[ID[2,t],:]$, we have

$$\begin{aligned}
RC[2] &= RC[1] + H[1,:] = H[1,:],\\
RC[3] &= RC[2] + H[2,:] = H[1,:] + H[2,:],\\
RC[4] &= RC[2] + H[3,:] = H[1,:] + H[3,:],\\
RC[5] &= RC[3] + H[3,:] = H[1,:] + H[2,:] + H[3,:],\\
RC[6] &= RC[3] + H[4,:] = H[1,:] + H[2,:] + H[4,:],\\
RC[7] &= RC[5] + H[4,:] = H[1,:] + H[2,:] + H[3,:] + H[4,:],\\
RC[8] &= RC[7] + H[5,:] = H[1,:] + H[2,:] + H[3,:] + H[4,:] + H[5,:].
\end{aligned}$$

Note that for any $1 \le t \le IDlen$, only one addition is needed to calculate $RC[t+1]$. Furthermore, we check whether the Rank of $RC[t+1]$ equals $n-1$ in each iteration of the loop.

Next, we will discuss the correspondence between $RC[t+1]$ and $CR[t]$ for any $1 \le t \le IDlen$. Note that if $CR[t] = \sum_{i=1}^{5} a_i \times 2^{i-1} \mapsto (a_1,a_2,a_3,a_4,a_5)$, then $RC[t+1] = \sum_{i=1}^{n} a_i H[i,:]$. For example

$$\begin{aligned}
&CR[1] = 1 \mapsto (1,0,0,0,0) \mapsto RC[2] = H[1,:],\\
&CR[2] = 3 \mapsto (1,1,0,0,0) \mapsto RC[3] = H[1,:] + H[2,:],\\
&CR[3] = 5 \mapsto (1,0,1,0,0) \mapsto RC[4] = H[1,:] + H[3,:],\\
&CR[4] = 7 \mapsto (1,1,1,0,0) \mapsto RC[5] = H[1,:] + H[2,:] + H[3,:],\\
&CR[5] = 11 \mapsto (1,1,0,1,0) \mapsto RC[6] = H[1,:] + H[2,:] + H[4,:],\\
&CR[6] = 15 \mapsto (1,1,1,1,0) \mapsto RC[7] = H[1,:] + H[2,:] + H[3,:] + H[4,:],\\
&CR[7] = 31 \mapsto (1,1,1,1,1) \mapsto RC[8] = H[1,:] + H[2,:] + H[3,:] + H[4,:] + H[5,:].
\end{aligned}$$

In a word, the smallest element from each cyclic equivalence set is saved in CR in ascending order. According to Corollary 1, we only need to calculate the rank of the nonzero linear combinations of the 5 rows of H correspond to $CR[t](1 \leq t \leq 7)$. We also note that $S_t \subset \mathbb{F}_2^n$ in Corollary 1. If all elements in S_t are converted to integers, then $CR[t]$ is the smallest element from this set.

Finally, we will explain why we need $ID = GetIndices(n)$ in our program. The ID stores some subscripts, which record how to iteratively obtain elements with larger subscripts from elements with smaller subscripts in ID. We need to generate CR first by $CR = getCyclicEquivalenceRepresentatives(n)$, then search for the iterative relationship between the preceding and following elements in CR which satisfies $CR[t] = CR[ID[1,t]] + 2^{ID[2,t]-1}(2 \leq t \leq 7)$. Based on the correspondence between $RC[t+1]$ and $CR[t]$ $(1 \leq t \leq 7)$ which have been listed above, we can obtain the ID with $RC[t+1] = RC[ID[1,t]+1]+H[ID[2,t],:]$ for any $1 \leq t \leq 7$ after adding the initial values of $ID[1,1]$ and $ID[2,1]$. For example, $CR[4] = 7 = CR[ID[1,4]]+2^{ID[2,4]-1} = CR[2]+2^{3-1} = 3+4$ implies $RC[5] = RC[ID[1,4]+1]+H[ID[2,4],:] = RC[2]+H[3,:]$.

4 Algorithm and experimental results

We will explain how to generate QAMs with the function IsQAM(n,H) in Algorithm 1.

4.1 Algorithm

The basic idea of the algorithm is to perform a depth-first search over all possible instantiations of the matrix H, verifying the necessary conditions for being a QAM at every step, and backtracking if they are violated. For the first unknown value in H, we can restrict our guesses to elements from $\mathcal{F}_n$. For subsequent unknowns, we have to consider all possible values from $\mathbb{F}_{2^n}^*$. Recall from the discussion above that knowing one value of H immediately allows us to deduce the values of all other entries on its diagonal. Thanks to this, the matrix has relatively few degrees of freedom, which makes the entire search feasible.

For clarity, we describe the algorithm for the specific case of $n = 5$. The same principle naturally extends to an arbitrary dimension n.

Suppose H is the same as in Example 1. According to Proposition 1, we only need to consider $a \in \mathcal{F}_5$. Then we can get Algorithm 2. We recall that $\mathcal{F}_n$ and $\mathbb{F}_{2^n}^*$ are defined in Definition 2.

Algorithm 2 is a special case of the algorithm in [26], except for Line 5. Algorithm 2 can be easily generalized to any n, and we will call them Algorithm 2 for $n = k$ $(k = 7, 8, 9, 10, 11)$ to denote such algorithms in the following. According to the results of Sections 3.1 and 3.2. Algorithm 2 for n is approximately $\frac{n \times 2^n}{n^3}$ times faster than the algorithm in [26]. Running Algorithm 2 for $n = 7, 8, 9$ on a computer with an Intel Core i7-7700 CPU

Input: A positive integer n, and a symmetric matrix $H \in \mathbb{F}_{2^n}^{n\times n}$ with $H[u,u]=0$, and $H[u+1,v+1]=H[u,v]^2$ for $1\le u,v\le n$;
Output: If H is QAM, return true; otherwise, return false.

1 **function** IsQAM(n,H):
2 ID=GetIndices(n);
3 IDlen=NumberOfColumns(ID);
4 RC=[ZeroVector]; // ZeroVector = $(0,0,\ldots,0) \in \mathbb{F}_{2^n}^n$
5 **for** $t \in [1..IDlen]$ **do**
6 rc=RC[ID[1,t]+1]+H[ID[2,t]];
7 Expand $rc \in \mathbb{F}_{2^n}^n$ to $MA \in \mathbb{F}_2^{n\times n}$;
8 **if** $GaussRank(MA) < n-1$ **then**
9 return false; // Caclculate the rank of MA with the Gaussian elimination method
10 **end**
11 Append rc to RC; // RC[t + 1] = RC[ID[1, t] + 1] + H[ID[2, t]]
12 **end**
13 return true;
14 **end function**

15 **function** GetIndices(n): // Return ID with CR[t] = CR[ID[1, t]] + $2^{\mathrm{ID}[2,\mathrm{t}]-1}$
16 $CR = getCyclicEquivalenceRepresentatives(n)$;
17 $len = \#CR$;
18 Initialize $ID \in \mathbb{Z}^{2\times len}$ with 0;
19 $ID[1,1]=0; ID[2,1]=1$;
20 **for** $t \in [2..len]$ **do**
21 $e = CR[t]$;
22 Select $j \in [1..n]$ with $2^{j-1} < e \le 2^j$;
23 Select $i \in [1..len]$ with $CR[i] == e - 2^{j-1}$; // We can always find such i since in Line 32 we select the smallest element in each cyclic equivalent class
24 $ID[1,t]=i; ID[2,t]=j$; // CR[t] = CR[ID[1, t]] + $2^{\mathrm{ID}[2,\mathrm{t}]-1}$
25 **end**
26 return ID;
27 **end function**

28 **function** getCyclicEquivalenceRepresentatives(n): // Return the smallest element in each cyclic equivalent class
29 CR=[ ];
30 NZ=$[t : t \in [1..2^n-1]]$;
31 **while** *true* **do**
32 $e = NZ[1]$; // Select the smallest element in NZ
33 Append e to the end of CR;
34 $k = e$;
35 **for** $i \in [1..n]$ **do**
36 Delete k from NZ; // Delete any element cyclic equivalent with e
37 $k = 2\times k$;
38 **if** $k > 2^n$ **then**
39 $k = k - 2^n + 1$;
40 **end**
41 **end**
42 **if** *NZ is empty* **then**
43 break;
44 **end**
45 **end**
46 return CR;
47 **end function**

Algorithm 1: Determine whether H is QAM

```
for a ∈ F_5 do
    if (0 a; a 0) is proper then
        for b ∈ F*_{2^5} do
            if (0 a b; a 0 a^2; b a^2 0) is proper then
                if IsQAM(5,H) then
                    Generate APN from H.
                end
            end
        end
    end
end
```

Algorithm 2: Generate APN functions on $\mathbb{F}_{2^5}$

at 3.6G GHz, it takes 14.5 seconds, 25 minutes and 4 days respectively to reproduce the results of the algorithm in [26]. Running Algorithm 2 for $n = 10$ on a server with an Intel Xeon 8336C 32C/64T CPU at 2.3 GHz, 22 Magma scripts in parallel, it takes about 100 days to get 29088 quadratic homogeneous APN functions with coefficients in $\mathbb{F}_2$. By computing the differential spectra of the ortho-derivative method [13] and then applying the equivalence algorithm from [9], these APN functions can be divided into 3 CCZ-inequivalent classes, all of which correspond to known APN functions. These are listed in Table 1.

Table 1 CCZ-inequivalent representatives for all quadratic APN functions with coefficients in $\mathbb{F}_2$ for $n = 10$

x^3
x^9
$x^{513} + x^{192} + x^{96} + x^{48} + x^{33} + x^{18} + x^{12}$

Despite not finding any new functions over $\mathbb{F}_{2^{10}}$, however, we found a shorter representative for $x^3 + \mathrm{Tr}(x^9)$. We can thus state that the CCZ-equivalence class of any quadratic APN functions over $\mathbb{F}_{2^n}$ for $n \leq 10$ with coefficients in $\mathbb{F}_2$. It was observed in [26] that any quadratic APN function F_1 with coefficients in $\mathbb{F}_2$ appears to be CCZ-equivalent to a quadratic APN function F_2 with at most n non-zero coefficients in $\mathbb{F}_{2^n}$ for $n \leq 9$. We see that the conjecture is true for $n = 10$.

4.2 Further observation

There are only 3 CCZ-inequivalent APN functions among the 29088 quadratic homogeneous APN functions on $\mathbb{F}_{2^{10}}$, which implies that a huge number of these functions are CCZ-equivalent to each other. Therefore, it is meaningful

to exclude some CCZ-equivalent functions in our algorithm. Note that in Algorithm 2 $a \in \mathcal{F}_5$ while $b \in \mathbb{F}_{2^5}^*$, and $\#\mathbb{F}_{2^5}^* \approx \#\mathcal{F}_5 \times 5$. Thus, it could make sense to restrict $b \in \mathcal{F}_5$ if doing a partial search. The pseudocode for such a partial search is given below as Algorithm 3. As before, we only describe the algorithm for the concrete case of $n = 11$; however, it naturally generalizes to any dimension n.

Generally, restricting the range of values for the left half of the first row's elements to $\mathcal{F}_n$ in Algorithm 2 for n, then running it on a computer with an Intel Core i7-7700 CPU at 3.6G GHz, we can get Table 2 . Inspired by Table 2, column 2 means that it takes 0.047 second to get 4 quadratic homogeneous APN functions on $\mathbb{F}_{2^6}$, others are similar, and these functions contain all CCZ-inequivalent quadratic APN functions with coefficients in $\mathbb{F}_2$ on $\mathbb{F}_{2^n}$ for $n = 6, 7, 8, 9, 10$.

Table 2 Time and APN functions

n	6	7	8	9	10
time	0.047s	0.375s	27.142s	582.781s	45 hours
count	4	78	142	26	45

Inspired by Table 2, we consider the case for $n = 11$. For clarity, let's briefly introduce the process of the algorithm. Let $H_{11} \in \mathbb{F}_{2^{11}}^{11\times 11}$. According to the definition of QAM and Theorem 1, if H_{11} is the corresponding QAM of a quadratic APN function with coefficients in $\mathbb{F}_2$, then we must have

$$H_{11}[1,:] = [0, a, b, c, d, e, e^{64}, d^{128}, c^{256}, b^{512}, a^{1024}],$$

for some $a, b, c, d, e \in \mathbb{F}_{2^{11}}$.

Assuming that a, b, c, d, e are unknown variables, let's investigate how to assign them to make H_{11} be QAM. A full search using algorithm 2 is unfortunately not feasible, so we instead

```
for a, b, c, d, e ∈ F_11 do
    if The k-th order submatrix of the upper left corner of H_11 is proper for
     k = 2, 3, 4, 5, 6 then
        if IsQAM(11, H_11) then
            Generate APN from H_11.
        end
    end
end
```

Algorithm 3: Partial search for APN functions on $\mathbb{F}_{2^{11}}$

Algorithm 3 can also be generalized to any n, and we will call it Algorithm 3 for n for short. We have realized it with Magma, and Table 2 is the results of Algorithm 3 for $n = 6, 7, 8, 9, 10$.

Algorithm 3 for $n = 11$ is about 11^4 times faster than Algorithm 2 for $n = 11$. For $n = 11$, running 15 Magma scripts in parallel on a server with an Intel Xeon 8336C 32C/64T CPU at 2.3 GHz for 12 days, we can obtain 29 APN functions. According to the differential and Walsh spectra of their ortho-derivatives, there are at least 6 CCZ-inequivalent classes among these functions, and these functions are all CCZ-equivalent to the known APN classes listed in Table 3. Unfortunately, none of the currently available algorithms for testing CCZ- or EA-equivalence can be applied to the these functions to get a complete classification of the 29 functions. We leave this as a problem for future work.

Table 3 CCZ-inequivalent representatives from the partial search for n = 11

x^3
x^5
x^9
x^{17}
x^{33}
$x^3 + \mathrm{Tr}(x^9)$

Nonetheless, we conjecture that Table 3 contains all CCZ-inequivalent quadratic APN functions with coefficients in $\mathbb{F}_2$ on $\mathbb{F}_{2^{11}}$, since Algorithm 3 for $n = 6, 7, 8, 9, 10$ can generate all CCZ-inequivalent quadratic APN functions with coefficients in $\mathbb{F}_2$ on $\mathbb{F}_{2^n}$ for $n = 6, 7, 8, 9, 10$.

5 Conclusion

We have introduced an algorithm which can speed up the algorithm in [26] by a factor of approximately $\frac{n \times 2^n}{n^3}$ times. With this algorithm, we performed an exhaustive search in dimension 10 and obtained that there are only 3 CCZ-inequivalent quadratic APN functions with coefficients in $\mathbb{F}_2$, all of which correspond to previously known APN functions. We have also found a shorter representative $x^{513}+x^{192}+x^{96}+x^{48}+x^{33}+x^{18}+x^{12}$ for $x^3+\mathrm{Tr}(x^9)$, meaning that any quadratic APN function over $\mathbb{F}_{2^n}$ with $n \leq 10$ and with coefficients in $\mathbb{F}_2$ is CCZ-equivalent to a function with at most n terms with non-zero coefficients. If we perform a partial search by restricting the values of elements in the matrix to a smaller set, we find 6 CCZ-inequivalent quadratic APN functions with coefficients in $\mathbb{F}_2$ over $\mathbb{F}_{2^{11}}$, and they are all CCZ-equivalent to known ones. We used the differential and Walsh spectra of the ortho-derivatives to split the functions into classes that are guaranteed to be CCZ-inequivalent to each other, and in the case of $n = 10$ we used the equivalence algorithm from [9] to verify that the functions within each of these classes are pairwise CCZ-equivalent. In the case of $n = 11$, we find 29 quadratic APN functions with coefficients in $\mathbb{F}_2$, which can be divided into 6 classes according to the differential and Walsh spectra of their ortho-derivatives. While it is not possible to verify that the functions within each of these classes are CCZ-equivalent

to each other using currently available tools, we can see that there are at least 6 CCZ-equivalence classes represented by these functions; and based on the accuracy of the Walsh and differential spectra of the ortho-derivatives as invariants, we can conjecture that there are precisely 6 CCZ-equivalence classes among these functions.

6 Acknowledgements

Yuyin Yu is supported by the National Key R&D Program of China (Grant No. 2021YFB3100200) and the GuangDong Basic and Applied Basic Research Foundation (Grant No. 2021A1515011904).

References

1. C. Beierle, M. Brinkmann, G. Leander. Linearly self-equivalent APN permutations in small dimension. IEEE Transactions on Information Theory, vol. 67, no. 7, pp. 4863–4875, 2021.
2. C. Beierle, G. Leander. New Instances of Quadratic APN Functions. IEEE Transactions on Information Theory, vol. 68, no. 1, pp. 670-678, 2022.
3. Beierle C., and Leander G.: New instances of quadratic APN functions in small dimension. Version 2.1, dataset, Zenodo, 2021, doi: 10.5281/zenodo.4738942.
4. C. Beierle, G. Leander, L. Perrin. Trims and extensions of quadratic APN functions. Designs, Codes and Cryptography, vol. 90, pp. 1009–1036, 2022.
5. Brinkmann M., Leander G.: On the classification of APN functions up to dimension five, Designs, Codes and Cryptography,vol. 49, no.1-3, pp. 273 - 288, 2008.
6. K. Browning, J. F. Dillon, M. McQuistan. APN polynomials and related codes. Special volume of Journal of Combinatorics, Information and System Sciences, vol. 34, pp. 135-159, 2009..
7. L. Budaghyan, M. Calderini, C. Carlet, R.S. Coulter, I. Villa. Constructing APN functions through isotopic shifts. IEEE Transactions on Information Theory, vol. 66, no. 8, pp. 5299–5309, 2020.
8. Budaghyan L., Ivkovic I., Kaleyski N.: Triplicate functions, Cryptography and Communications, 15: 35–83, 2023.
9. Ivkovic I, Kaleyski N.: Deciding and reconstructing linear equivalence of uniformly distributed functions. Cryptology ePrint Archive. 2022.
10. Budaghyan L., Carlet C.: Classes of quadratic APN trinomials and hexanomials and related structures, IEEE Trans. Inf. Theory, vol. 54, no. 5, pp. 2354-2357, 2008.
11. Budaghyan L., Carlet C., Leander G.: Two classes of quadratic APN binomials inequivalent to power functions, IEEE Transactions on Information Theory, vol. 54, no. 9, pp. 4218-4229, 2008.
12. Budaghyan L., Helleseth T., and Kaleyski N.: A new family of APN quadrinomials, IEEE Transactions on Information Theory, vol. 66, no. 11, pp. 7081–7087, Nov. 2020.
13. A. Canteaut, A. Couvreur, L. Perrin. Recovering or Testing Extended-Affine Equivalence. IEEE Transactions on Information Theory, vol. 68, no. 9, pp. 6187-6206, 2022.
14. C. Carlet. Boolean functions for cryptography and coding theory. Cambridge ; New York, NY : Cambridge University Press, 2020.
15. Carlet C., Charpin P., Zinoviev V.: Codes, bent functions and permutations suitable for DES-like cryptosystems, Designs, Codes and Cryptography, 15(2):125-156, 1998.
16. Y. Edel. Quadratic APN functions as subspaces of alternating bilinear forms. In: Proceedings of the Contact Forum Coding Theory and Cryptography III, Belgium 2009, pp. 11–24, 2011.

17. Y. Edel, G. Kyureghyan, A. Pott. A new APN function which is not equivalent to a power mapping. IEEE Transactions on Information Theory, vol. 52, no. 2, pp. 744-747, 2006.
18. Y. Edel, A. Pott. A new almost perfect nonlinear function which is not quadratic. Advances in Mathematics of Communications, vol. 3, no. 1, pp. 59–81, 2009.
19. K. Kalgin, V. Idrisova. The classifcation of quadratic APN functions in 7 variables and combinatorial approaches to search for APN functions. Cryptography and Communications, vol. 15, pp. 239–256, 2023.
20. Gold R.: Maximal recursive sequences with 3-valued recursive cross correlation functions, IEEE Transactions on Information Theory, vol. IT-14, no. 1, pp. 154–156, Jan. 1968.
21. H. Taniguchi. On some quadratic APN functions. Designs, Codes and Cryptography, vol. 87, no. 9, pp. 1973–1983, 2019.
22. Perrin L.: How to take a function apart with SboxU, The 5th International Workshop on Boolean Functions and their Applications (BFA 2020) (2020).
23. G. Weng, Y. Tan, G. Gong. On quadratic almost perfect nonlinear functions and their related algebraic object, in Proc. Int. Workshop Coding Cryptogr. (WCC), 2013.
24. S. Yoshiara. Equivalence of quadratic APN functions. Journal of Algebraic Combinatorics, vol. 35, pp. 461-475, 2012.
25. Y. Yu, L. Perrin. Constructing more quadratic APN functions with the QAM method. Cryptography and Communications, vol. 14, no. 6, pp. 1359-1369, 2022.
26. Y. Yu, N. Kaleyski, L. Budaghyan, Y. Li. Classification of quadratic APN functions with coefficients in F2 for dimensions up to 9. Finite Fields and Their Applications, vol. 68, Art. no. 101733, 2020.
27. Y. Yu, M. Wang, Y. Li. A matrix approach for constructing quadratic APN functions. IACR Cryptol. ePrint Arch., Tech. Rep. 2013/007, 2013.
28. Y. Yu, M. Wang, Y. Li. A matrix approach for constructing quadratic APN functions. Designs, Codes and Cryptography, vol. 73, pp. 587-600, 2014.
29. Lijing Z., Haibin K., Yanjun L., Jie P., Deng T.: Constructing new APN functions through relative trace functions, IEEE Transactions on Information Theory, vol. 68, no. 11, pp. 7528–7537, Nov. 2022.
30. W. Cannon, John amd Bosma, C. Fieker, and A. Steel, Handbook Magma Functions, 2nd ed. 2014. [Online]. Available: https://www.math.uzh.ch/sepp/magma-2.20.4-cr/