Some Experimental Results on Quadratic APN Functions

Shibam Ghosh[1,2] and Léo Perrin[2]

[1]University Of Haifa, Haifa, Israel
[2]Inria, France

Abstract

The Big APN Problem is the still ongoing search for APN permutations operating on an number n of bits that is even and such that $n \geq 8$. An APN function F is a vectorial Boolean function such that $F(x + a) - F(x) = b$ has at most 2 solutions whenever $a \neq 0$.

This note presents our attempt at generating quadratic APN functions for $n \geq 8$. The core idea consists in representing a quadratic vectorial Boolean function using a cubic structure called quadratic indicator cube (QIC). While our definition of this object is very simple, parallel are drawn with the Jacobian operator. The QIC is defined for any quadratic function, but it is easy to figure out if the corresponding function is APN by computing the rank of several binary matrices deduced from the QIC.

Then we present some algorithms based on backtracking to change the elements of the QIC in such a way that, if we start from an APN function, then it remains APN. However, despite exploring large spaces, our algorithm fails to return new APN functions for $n = 8$. This hints at a low density of the set of APN functions within the set of quadratic functions, and at a large distance between them.

1 Introduction and Definitions

Let $\mathbb{F}_2 = \{0, 1\}$ be the finite field with 2 elements. Vectorial boolean functions are functions mapping $\mathbb{F}_2^m$ to $\mathbb{F}_2^n$ for some strictly positive integers m and n. Such functions appear, among other cases, in symmetric cryptography where they can be used for instance either as *S(ubstitution)-boxes* or as components of the filter function of a stream cipher.

In such contexts, one key property of a vectorial boolean function $F : \mathbb{F}_2^m \rightarrow \mathbb{F}_2^n$ is its *differential uniformity* [9]. It is the maximum number of solution of the equation

$$F(x + a) - F(x) = b \ , \tag{1}$$

for all non-zero input differences a. If this quantity is low, then using F in a symmetric primitive will contribute to make it resistant to differential cryptanalysis [2].

1.1 The Big APN Problem

The minimum value of the differential uniformity is 2 in characteristic 2. Indeed, if $x \in \mathbb{F}_2^m$ is a solution of Equation (1), then so is $x - a$. Functions reaching this minimum are called *Almost Perfect Non-linear (APN)*, and have been the topic of intense scrutiny over the past 30 years. In particular, for practical reasons, the most useful such functions would be APN permutations operating on an even number of bits, e.g. $m = n = 8$. However, we still do not know if APN permutations operating on an even number of variables greater than 8 even exist. This problem is nick-named the *Big APN Problem.* The most notable progress in this area occurred when Dillon et al. found a 6-bit APN permutation [4], but this result could not be generalized.

The approach of Dillon et al. consisted in finding a 6-bit quadratic function, and then transforming it into a permutation in a way that preserves differential uniformity. This transformation was based on *CCZ-equivalence* [7], the definition of which is not needed here (we refer the interested reader to [7] and [6] for more information).

In this work, our aim was to emulate this work. Thanks to the framework established in [6] and to the algorithm presented in [3], it is now trivial to check whether there exists a permutation that is CCZ-equivalent to a given function. As a consequence, a tempting approach to find APN permutations consists in generating as large a set of APN functions as possible, and then to apply these tools to check if one of them happens to be CCZ-equivalent to a permutation.

This direction has been followed by [10] and, more recently, by [1]. In both cases, the authors designed dedicated algorithms and implemented them, thus generating thousands of new quadratic APN functions operating on 8 bits (or more).

A more complete description of this work can be found in the master thesis of the first author [8].

Our method relies on a tool we called *Quadratic Indicator Cube (QIC)*. It is presented in Section 2, along with algorithms allowing the generation of new APN functions starting from a known one. We discuss our results in Section 3; but first let us introduce the necessary mathematical concepts.

1.2 Notations and Algebraic Normal Form

As mentionned before, we let $\{0,1\} = \mathbb{F}_2$ be the finite field with two elements. The canonical basis of $\mathbb{F}_2^n$ is denoted $\{e_i\}_{0 \leq i < n}$. The Hamming weight of $x \in \mathbb{F}_2^n$ is the number of indices i such that $x_i = 1$, and is denoted $\mathrm{hw}(x)$. In what follows, we restrict ourselves to functions mapping $\mathbb{F}_2^n$ to itself (i.e. the input and output sizes are the same). We let $\Delta_a F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be the *derivative* of F along $a \in \mathbb{F}_2^n$, where $\Delta_a F(x) = F(x+a) + F(x)$.

Each Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ has a uniquely defined *Algebraic Normal Form (ANF)* which is such that

$$f(x_0, ..., x_{n-1}) = \sum_{u \in \mathbb{F}_2^n} a_u^f x^u \ ,$$

where $a_u^f \in \mathbb{F}_2$ for all $u \in \mathbb{F}_2^n$, and where $x^u = \prod_{i=0}^{n-1} x_i^{u_i}$. The *algebraic degree* of f is the maximum Hamming of all u such that $a_u^f = 1$. For a vectorial Boolean function $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, the algebraic degree is the maximum degree of its components. A *quadratic* function has algebraic degree 2.

Finally, we let QH_n be the set of all quadratic homogenous mapping $\mathbb{F}_2^n$ to itself. It contains all functions $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ such that the ANF of each of their coordinate F_i satisfies $a_u^{F_i} = 1 \implies \mathrm{hw}(u) = 2$.

2 The Quadratic Indicator Cube

Definition 2.1 (Quadratic Indicator Cube (QIC)) *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a function of* QH_n *with coordinates $(F_0, ..., F_{n-1})$. We call* Quadratic Indicator Cube (QIC) *the 3-dimensionnal array of elements $Q_{i,j}^k \in \mathbb{F}_2$ such that*

$$F_k(x_0, ..., x_{n-1}) = \sum_{0 \leq i < j < n} Q_{i,j}^k x_i x_j \ ,$$

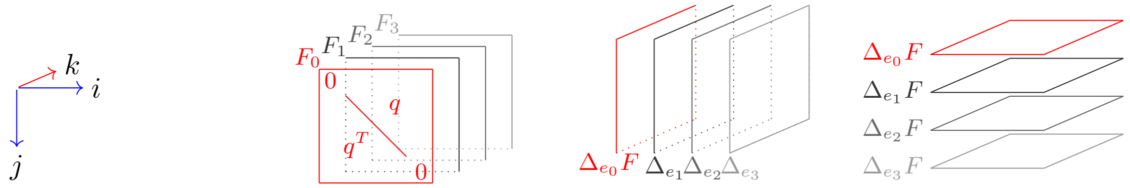
and such that $Q_{i,j}^k = Q_{j,i}^k$.

2.1 Basic Properties

It is easy to express derivatives of a function $F \in \mathrm{QH}_n$ using its QIC:

$$\Delta_{e_i} F_k(x) = \sum_{j=0}^{n-1} Q_{i,j}^k x_j \ . \qquad (2)$$

In other words, it is possible to evaluate the derivative of F along a specific canonical basis vector using a matrix multiplication. Figure 1 presents various ways of interpreting the QIC of a function of QH_n.



(a) Orientation. (b) Coordinates. (c) Derivatives (e_i). (d) Derivatives (e_j).

Figure 1: The meaning of the different parts of the QIC.

A priori, it is tempting to think that the QIC is only related to the derivatives along the vectors of the canonical basis. However, in the case of quadratic homogenous functions, these derivatives can be used to express all derivatives (including those *not* along canonical basis vectors). This is stated in the following theorem.

Theorem 2.2 *Let $F \in \mathrm{QH}_n$. Then it holds that*

$$\Delta_a F(x) \;=\; F(a) + \sum_{i=0}^{n-1} a_i \Delta_{e_i} F(x)$$

Its proof relies on the linearity of the derivatives of all functions of QH_n, as well as on the fact that, for any function (including non-quadratic ones),

$$\Delta_a F(x) \;=\; F(a) + \sum_{i=0}^{n-1} a_i \Delta_{e_i} F\Big(x + \sum_{j=0}^{n-1} a_j e_j\Big) \;;$$

this being proved via an induction on the Hamming weight of a. By combining Theorem 2.2 with Equation (2), we obtain the following corollary.

Corollary 2.3 *For any $F \in \mathrm{QH}_n$ and any $a \in \mathbb{F}_2^n$, we have*

$$\mathrm{Im}(\Delta_a F) \;=\; F(a) + \left\{ \sum_{i=0}^{n-1} a_i \begin{bmatrix} Q_{i,0}^0 & Q_{i,1}^0 & \dots & Q_{i,n-1}^0 \\ \dots & \dots & & \dots \\ Q_{i,0}^{n-1} & Q_{i,1}^{n-1} & \dots & Q_{i,n-1}^{n-1} \end{bmatrix} \begin{bmatrix} x_0 \\ \dots \\ x_{n-1} \end{bmatrix}, x \in \mathbb{F}_2^n \right\} ,$$

and consequently

$$\dim\big(\mathrm{Im}(\Delta_a F)\big) \;=\; \mathrm{rank}\left(\sum_{i=0}^{n-1} a_i \begin{bmatrix} Q_{i,0}^0 & Q_{i,1}^0 & \dots & Q_{i,n-1}^0 \\ \dots & \dots & & \dots \\ Q_{i,0}^{n-1} & Q_{i,1}^{n-1} & \dots & Q_{i,n-1}^{n-1} \end{bmatrix} \right) .$$

The following corollary follows directly.

Corollary 2.4 *A function $F \in \mathrm{QH}_n$ is APN if and only if, for all $a \in \mathbb{F}_2^n$, we have*

$$\mathrm{rank}\left(\sum_{i=0}^{n-1} a_i \begin{bmatrix} Q_{i,0}^0 & Q_{i,1}^0 & \dots & Q_{i,n-1}^0 \\ \dots & \dots & & \dots \\ Q_{i,0}^{n-1} & Q_{i,1}^{n-1} & \dots & Q_{i,n-1}^{n-1} \end{bmatrix} \right) \;=\; n-1 \;.$$

Interestingly, in [5], it is shown that the linear part of the Jacobian of a quadratic function has the exact same behaviour as the QIC. The Jacobian $\mathrm{Jac}F$ at $x \in \mathbb{F}_2^n$ of a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^n$ is the parameterised matrix defined by

$$\mathrm{Jac}F(x) \;=\; \begin{bmatrix} \Delta_{e_0} F_0(x) & \dots & \Delta_{e_{n-1}} F_0(x) \\ \dots & & \dots \\ \Delta_{e_0} F_{n-1}(x) & \dots & \Delta_{e_{n-1}} F_{n-1}(x) \end{bmatrix} ,$$

and for a function in QH_n it holds that $\Delta_a F(x) + F(a) = \mathrm{Jac}F(a) \times x$. Note that the entries in the matrix $\mathrm{Jac}F(a)$ correspond to the quantity evaluated in Equation (2). In other words, it holds that.

$$\mathrm{Jac}F(a) \;=\; \sum_{i=0}^{n-1} a_i \begin{bmatrix} Q_{i,0}^0 & Q_{i,1}^0 & \dots & Q_{i,n-1}^0 \\ \dots & \dots & & \dots \\ Q_{i,0}^{n-1} & Q_{i,1}^{n-1} & \dots & Q_{i,n-1}^{n-1} \end{bmatrix}$$

2.2 Modifying the QIC

Starting from the QIC of an APN function, it is possible to devise an algorithm that modifies in such a way that it remains the QIC of an APN function—one that is hopefully different from the starting point. We investigated several approaches.

Each time, our approach consists in removing all values $Q^k_{i,j}$ of the QIC for some specific set of coordinates (i, j, k). Then, these are replaced one by one in such a way as to ensure that the rank condition specified in Corollary 2.4 still holds. We also need to retain the property that $Q^k_{i,j} = Q^k_{j,i}$.

The inner workings of such an algorithm are non-trivial to implement, and performances play a critical role since the search space is large. The main question is then the choice of the set of coordinates (i, j, k) to remove, and the order in which the replacing values should be tried. In particular, some orders allow the precomputation of valid candidates for each entry independently from the rest of the computation.

2.2.1 Modifying one coordinate

In this case, we remove all entries $Q^0_{i,j}$ from the QIC of an APN function of QH_n. Recall that the matrix $\{Q^k_{i,j}\}_{0\leq j<n,0\leq k<n}$ corresponds to $\Delta_{e_i}F$ (see Figure 1c), meaning that it has to be of rank $n-1$ along with all of its linear combinations with $\Delta_{e_{i'}}$ for $i' \neq i$. As a consequence, we can precompute for each i the set V_i of all u such that replacing the entries $Q^0_{i,j}$ with u_j for all j gives a new matrix $\{Q^k_{i,j}\}_{0\leq j<n,0\leq k<n}$ which differs on its first column, and which has rank $n-1$. The algorithm then uses a basic backtracking approach to find a valid sequence $\{v_0, ..., v_{n-1}\}$ such that

- the resulting 3D object Q' is symmetric, and
- the rank condition of Corollary 2.4 is satisfied.

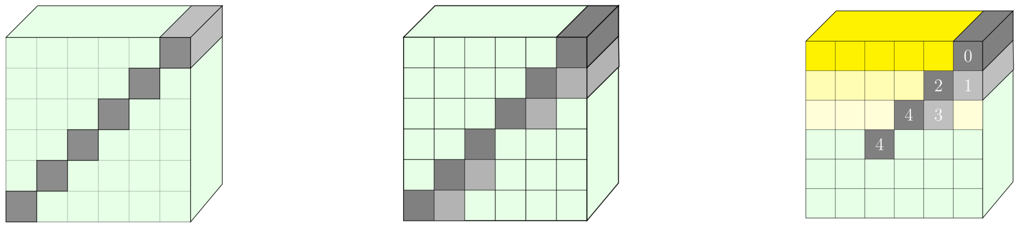
This verification is done at each step, which allows a significant speed up. Unfortunately, this algorithm rarely returns any new function, and we could not find such an instance for $n = 8$ (despite exploring the full search space). Modifying two coordinates at the same time turned out to be too much from a computational standpoint, and we could not find even negative results in this case for $n = 8$.

2.2.2 Modifying two diagonals

Another approach consists in removing the elements with coordinates $(n-i-1, i, k)$, i.e on the anti-diagonal (see Figure 2a). To increase the size of the search space, we in fact remove two sub-diagonals (see Figure 2b). The advantage of this strategy is that, one guessed in the right order, we can have substantial filtering at each step of the process. This is illustrateged in Figure 2c. The guess with index 0 (which corresponds to $\{Q_{n-1,0,k}\}_{0\leq k<n}$) is constrained by the fact that the horizontal plane $j = 0$ (which corresponds to $\Delta_{e_0}F$) must have rank $n-1$. Once this quantity is known, in the next step, we can guess the element at position 1. In this case, the contraint comes from the fact that, once this vector is specified, the full derivative $\Delta_{e_{n-1}}$ is specified. Thus, we can check if this derivative has the appropriate rank, and if $\Delta_{e_0+e_{n-1}}$ does as well (which increases the pruning of this tree-based search). This process is iterated as summarized in Figure 2c until the last guess is successfully identified.

Using this method, we can quickly generate all 13 quadratic APN functions operating on 6 bits starting e.g. from the QIC of the cube mapping. This implies in particular that it is capable of generating functions with a different Walsh spectrum as the starting point.

Unfortunately, as for the coordinate modification method, this one does not return any new function for $n = 8$. Thanks to our efficient implementation, the algorithm finishes (i.e., explores all valid guesses) in less than one hour; but it does not return any function that is not the original one.



(a) The anti-diagonal. (b) Two anti-diagonals. (c) Guessing order.

Figure 2: The parts of the QIC that we modify (recall that the main diagonal is filled with zeroes).

3 Discussion

We have introduced the QIC a new tool to manipulate quadratic (APN) functions, and developed algorithms that rely on it to generate new APN functions starting from a known one. These new functions would have ANF related to the starting function, but differ in sufficiently many places that it is possible to generate all 6-bit quadratic APN functions starting from a unique one. However, the fact that our methods cannot[1] generate new functions for $n = 8$ indicates that the set of quadratic APN function has a very low density within the set of all quadratic functions, and that finding an APN function with a given ANF does not simplify finding functions that are "similar" to it.

It remains to be seen if different guessing strategies could lead to better results. In particular, an increase of the search space (i.e., erasing and replacing more entries in the QIC) coupled with a heuristic to explore it perhaps not exhaustively but faster could lead to the identification of new quadratic APN functions starting from a known one.

References

[1] Christof Beierle and Gregor Leander. New instances of quadratic APN functions. *CoRR*, abs/2009.07204, 2020.

[2] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4(1):3–72, January 1991.

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

[4] 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.

[5] Anne Canteaut, Alain Couvreur, and Léo Perrin. Recovering or testing extended-affine equivalence. Cryptology ePrint Archive, Report 2021/225, 2021. `https://eprint.iacr.org/2021/225`.

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

[7] 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.

[1] Due to our implementation, we insist that this failure is caused by the *absence* of solution, not by the fact that finding a solution would be computationally infeasible.

[8] Shibam Ghosh. On the QIC of quadratic APN functions. Master thesis defended at Inria, Paris, 2020.

[9] Kaisa Nyberg. Differentially uniform mappings for cryptography. In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 55–64. Springer, Heidelberg, May 1994.

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