Applicable Algebra in Engineering, Communication and Computing
https://doi.org/10.1007/s00200-022-00575-2

ORIGINAL PAPER

On cryptographic properties of cubic and splitting Boolean functions

Augustine Musukwa[1] · Massimiliano Sala[2] · Irene Villa[2] · Marco Zaninelli[2]

Received: 9 April 2020 / Revised: 5 July 2022 / Accepted: 16 July 2022

Abstract
The weight, balancedness and nonlinearity are important properties of Boolean functions, but they can be difficult to determine in general. In this paper, we study how to compute them for two classes of functions where these problems are more tractable. In particular, we study functions of degree three and the so-called "splitting" functions. The latter are functions that can be written as the sum of two functions defined over disjoint sets of variables. We show how, for splitting functions, studying these properties reduces to the study of simpler functions. We provide then a procedure to compute the weight of a cubic Boolean function. We show computationally that, for a cubic Boolean function with limited number of terms, this procedure is on average significantly more efficient than some other methods.

Keywords Boolean functions · Cubic Boolean functions · Nonlinearity · Weight

Mathematics Subject Classification 06E30 · 94A60 · 14G50

✉ Irene Villa
irene1villa@gmail.com

Augustine Musukwa
augustinemusukwa@gmail.com

Massimiliano Sala
maxsalacodes@gmail.com

Marco Zaninelli
zaninelli.marco21@gmail.com

[1] Mzuzu University, P/Bag 201, Mzuzu 2, Luwinga, Malawi

[2] University of Trento, Via Sommarive, 14, 38123 Povo, Trento, Italy

Published online: 06 August 2022

1 Introduction

Boolean functions are widely studied and they have applications in coding theory, cryptography and other fields. In cryptography, the properties of (vectorial) Boolean functions play a critical role, particularly when these functions are involved in the design of symmetric-key algorithms, such as block ciphers (in S-boxes) and stream ciphers (in nonlinear filters and combiners). When designing a cryptosystem, we want it to resist most of the known attacks. For this reason, a lot of effort is required to find a Boolean function with good cryptographic properties.

Various criteria can be used to measure the ability of Boolean functions to resist some cryptanalysis. For instance, balancedness, high nonlinearity and good autocorrelation properties provide good resistance to linear cryptanalysis and differential cryptanalysis [8]. It is not easy to find a function satisfying many such criteria at once, but usually we try to achieve a reasonable compromise by focusing on a few particular properties. In this paper, we study the weight, balancedness and nonlinearity of a particular class of Boolean functions that we call "splitting" functions. We study the same properties also for cubic Boolean functions that are not necessarily splitting functions. By "splitting" functions we refer to those Boolean functions that can be written as the sum of two Boolean functions defined over disjoint sets of variables, that is, $f(x_1, \ldots, x_n) = g(x_1, \ldots, x_s) + h(x_{s+1}, \ldots, x_n)$.

This paper is organised as follows. Sect. 2 reports some known preliminary results. In Sect. 3, we show how the weight of any Boolean function can be related to the weights of some other functions with lower dimension. In addition, we prove some results on the weight and balancedness of splitting functions, and of a special class of cubic Boolean functions. This allows us to present a procedure for computing the weight of a (generic) cubic Boolean function. In Sect. 4, we present an inequality which relates the nonlinearity of any Boolean function to the nonlinearity of some other functions of lower dimension. Finally, we compute the nonlinearity of splitting functions with a given shape.

2 Preliminaries

In this section we report some definitions and results which we use in our work. For more details, the reader is referred to [1, 2, 4, 5, 7, 9].

The symbol $\mathbb{N}$ denotes the set of natural numbers. Throughout the paper, unless otherwise specified, n denotes a positive integer. We denote the field of two elements by $\mathbb{F}$, and the vector space of dimension n over $\mathbb{F}$, for $n \in \mathbb{N}$, by $\mathbb{F}^n$. The vectors in $\mathbb{F}^n$ denoted by 0_n and 1_n are the vectors whose entries are, respectively, all zeros and all ones. Given a set A, $|A|$ denotes its size.

A *vectorial Boolean function (vBf)* is any function F from $\mathbb{F}^n$ to $\mathbb{F}^m$, for some positive integers n, m. A *Boolean function (Bf)* is any function f from $\mathbb{F}^n$ to $\mathbb{F}$, for some $n \geq 1$. Thus, Boolean functions are vectorial Boolean functions with $m = 1$. A vBf can be viewed as a concatenation of Bf's. Indeed, we can write a vBf as $F = (f_1, \ldots, f_m)$, where the Bf's $f_1, \ldots, f_m$ are called the *coordinate functions* of F.

In the present paper, we focus on Boolean functions. For $n \geq 1$, we denote by B_n the set of all Boolean functions from $\mathbb{F}^n$ to $\mathbb{F}$. For $1 \leq k < n$, if f is in B_n and depends only on k variables, we denote by $f_{\restriction \mathbb{F}^k}$ its restriction to these k variables. Clearly, $f_{\restriction \mathbb{F}^k}$ is in B_k.

Note Consider $1 \leq k < n$. To simplify the notation, sometimes for a Boolean function $g \in B_n$ we write $g(x_1, \ldots, x_k)$ to indicate that g depends only on k variables.

We use the algebraic normal form (ANF for short) to represent Bf's. This representation is unique and, given $f \in B_n$, it is the n-variate polynomial representation over $\mathbb{F}$ given by

$$f(x_1, \ldots, x_n) = \sum_{I \subseteq \mathcal{P}} a_I \left(\prod_{i \in I} x_i \right),$$

where $\mathcal{P} = \{1, \ldots, n\}$ and $a_I \in \mathbb{F}$. When $a_I \neq 0$, the element $\prod_{i \in I} x_i$ is called a *term* of f. The *algebraic degree* or simply *degree* of f can be defined as the value

$$\deg(f) = \max_{\substack{I \subseteq \mathcal{P} \\ a_I \neq 0}} |I|.$$

We say that f is *linear* if $\deg(f) \leq 1$ and $f(0) = 0$, *affine* if $\deg(f) \leq 1$, *quadratic* if $\deg(f) \leq 2$ and *cubic* if $\deg(f) \leq 3$.

Consider $f \in B_n$, for a positive integer n. The *Hamming weight* of f is given by $\mathrm{w}(f) = |\{x \in \mathbb{F}^n \mid f(x) = 1\}|$, and we say that f is *balanced* if $\mathrm{w}(f) = 2^{n-1}$. All non-constant affine functions are balanced. The *distance* between f and g is $d(f, g) = \mathrm{w}(f + g)$ and the *nonlinearity* of f is $\mathcal{N}(f) = \min_{\alpha \in A_n} d(f, \alpha)$, where A_n is the set of all affine Boolean functions in n variables.

The *Walsh transform* of f is the function $\mathcal{W}_f$ from $\mathbb{F}^n$ to $\mathbb{Z}$, defined as

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

for all $a \in \mathbb{F}^n$ and where "$\cdot$" is any inner product in $\mathbb{F}^n$. We define $\mathcal{F}(f)$ as

$$\mathcal{F}(f) = \mathcal{W}_f(0) = \sum_{x \in \mathbb{F}^n} (-1)^{f(x)} = 2^n - 2\mathrm{w}(f).$$

Observe that f is balanced if and only if $\mathcal{F}(f) = 0$.

The nonlinearity of a Bf f can also be expressed as $\mathcal{N}(f) = 2^{n-1} - \frac{1}{2}\mathcal{L}(f)$, where $\mathcal{L}(f) = \max_{a \in \mathbb{F}^n} |\mathcal{W}_f(a)|$. A Bf f on n variables is called *bent* if $\mathcal{N}(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$ (this can only happen for n even). The lowest possible value of $\mathcal{L}(f)$ is $2^{\frac{n}{2}}$, and the bent functions are precisely those that meet this bound with equality. For n odd, a Bf f is called *semi-bent* if $\mathcal{N}(f) = 2^{n-1} - 2^{\frac{n-1}{2}}$.

Let $a \in \mathbb{F}^n$. The *first-order derivative*, or simply the derivative, of $f \in B_n$ in the direction of a is defined by

$$D_a f(x) = f(x+a) + f(x).$$

The following result is well known, see for instance [4] Theorem 12.

Theorem 1 *A Bf f on n variables is bent if and only if $D_a f$ is balanced for any nonzero $a \in \mathbb{F}^n$.*

Two Bf's $f, g \in B_n$ are said to be *affine equivalent* if there exists an affine automorphism $\varphi : \mathbb{F}^n \to \mathbb{F}^n$ such that $f = g \circ \varphi$. This relation is denoted by $\sim_A$ and we write $f \sim_A g$. Observe that $\sim_A$ is an equivalence relation. The following results are well known, for example they can be derived from Proposition 13 in [4].

Proposition 2 *Let $f, g \in B_n$ be such that $f \sim_A g$. Then $\mathrm{w}(f) = \mathrm{w}(g)$ and so f is balanced if and only if g is balanced.*

Remark 3 Since, by Proposition 2, $\mathrm{w}(f) = \mathrm{w}(g)$ if $f \sim_A g$, this also implies that $\mathcal{F}(f) = \mathcal{F}(g)$ as $\mathcal{F}(f) = 2^n - 2\mathrm{w}(f)$.

Proposition 4 *Let $f, g \in B_n$ be such that $f \sim_A g$. Then*

$$\{|\mathcal{W}_f(a)|\}_{a \in \mathbb{F}^n} = \{|\mathcal{W}_g(a)|\}_{a \in \mathbb{F}^n}.$$

Moreover, we have $\mathcal{N}(f) = \mathcal{N}(g)$.

Remark 5 Therefore, the nonlinearity, the weight and the balancedness are affine invariants. Moreover, we have that the algebraic degree is also an affine invariant.

Next we present a well-known theorem on classification of quadratic Boolean functions, see [7] page 438. This representation of quadratic Bf's is sometimes called *Dickson form*. Indeed, Dickson calculated explicitly the Hamming weight of quadratic functions, by showing that any non-affine quadratic Boolean function $f \in B_n$ is affine equivalent to $x_1x_2 + \cdots + x_{2k-1}x_{2k} + cx_{2k+1} + d$, with $c, d \in \mathbb{F}$ and $2k \leq n$ (if $c = 1$, then $2k + 1 \leq n$).

Theorem 6 *Let $f \in B_n$ be quadratic but not affine. Then*

(i) $f \sim_A x_1x_2 + \cdots + x_{2k-1}x_{2k} + x_{2k+1}$ *with* $k \leq \lfloor \frac{n-1}{2} \rfloor$, *if f is balanced* ($\mathrm{w}(f) = 2^{n-1}$),

(ii) $f \sim_A x_1x_2 + \cdots + x_{2k-1}x_{2k} + c$, *with* $k \leq \lfloor \frac{n}{2} \rfloor$ *and* $c \in \mathbb{F}$, *if f is not balanced.*

In this second case we have that if $c = 0$ then $\mathrm{w}(f) < 2^{n-1}$, and if $c = 1$ then $\mathrm{w}(f) > 2^{n-1}$.

The proofs of the next theorem and lemma can be found in [6] page 134.

Theorem 7 *Let f be a quadratic Bf denoted as in Theorem 6. Then we have $\mathcal{W}_f(a) \in \{0, \pm 2^{n-k}\}$, for $a \in \mathbb{F}^n$, and $\mathcal{N}(f) = 2^{n-1} - 2^{n-k-1}$.*

Moreover, if f is not balanced then $\mathrm{w}(f) = \mathcal{N}(f) = 2^{n-1} - 2^{n-k-1}$ *if* $c = 0$, *and* $\mathrm{w}(f) = 2^n - \mathcal{N}(f) = 2^{n-1} + 2^{n-k-1}$ *if* $c = 1$.

Remark 8 Note that, for n even and $k = \frac{n}{2}$, f in Theorem 6 is bent. Obviously, this cannot happen for balanced functions.

Lemma 9 *Two quadratic Bf's g and h on $\mathbb{F}^n$ are affine equivalent if and only if* $\mathrm{w}(g) = \mathrm{w}(h)$ *and* $\mathcal{N}(g) = \mathcal{N}(h)$.

The following result is well known.

Proposition 10 *A Bf $g(x_1, \ldots, x_{n-1}) + x_n$ on n variables is balanced, for any $g \in B_{n-1}$.*

3 On the weight of Boolean functions

In this section we study Boolean functions of a particular form that we call *splitting functions*. Splitting functions can be characterised by means of Boolean functions of lower dimensions. By studying properties of these latter functions, we can determine important properties of the splitting function.

In particular, in this section we classify the weight of splitting functions and the weight of Boolean functions of degree three. Moreover, we determine some conditions for the balancedness of these functions.

Definition 11 A Bf f on n variables is a *splitting function* if

$$f \sim_A g(x_1, \ldots, x_s) + h(x_{s+1}, \ldots, x_n)$$

for some positive integer $s < n$. We recall that with this notation we assume that $g \in B_n$ depends only on s variables and $h \in B_n$ depends only on $n - s$ variables.

Remark 12 If we consider a Boolean function $g(x_1, \ldots, x_s) \in B_n$, with $s < n$ positive integers, then $\mathrm{w}(g) = 2^{n-s}\mathrm{w}(g_{\restriction \mathbb{F}^s})$ and $\mathcal{F}(g) = 2^{n-s}\mathcal{F}(g_{\restriction \mathbb{F}^s})$. Furthermore, g is balanced if and only if $g_{\restriction \mathbb{F}^s}$ is balanced and also $\mathcal{F}(g) = 0$ if and only if $\mathcal{F}(g_{\restriction \mathbb{F}^s}) = 0$.

We study here the weight and balancedness of splitting Bf's.

Theorem 13 *Let $f \in B_n$ be such that $f \sim_A g(x_1, \ldots, x_s) + h(x_{s+1}, \ldots, x_n)$, with $s < n$. Then*

$$\mathcal{F}(f) = \mathcal{F}(g_{\restriction \mathbb{F}^s})\mathcal{F}(h_{\restriction \mathbb{F}^{n-s}}) = 2^{-n}\mathcal{F}(g)\mathcal{F}(h).$$

Proof Since, by Remark 3, $\mathcal{F}(f)$ is invariant under affine equivalence, we have

$$\begin{aligned}\mathcal{F}(f) &= \sum_{(y,x)\in\mathbb{F}^s\times\mathbb{F}^{n-s}} (-1)^{g(y)+h(x)} = \sum_{y\in\mathbb{F}^s}(-1)^{g(y)} \sum_{x\in\mathbb{F}^{n-s}} (-1)^{h(x)} \\ &= \mathcal{F}(g_{\restriction\mathbb{F}^s})\mathcal{F}(h_{\restriction\mathbb{F}^{n-s}}) = 2^{-n}\left(2^{n-s}\mathcal{F}(g_{\restriction\mathbb{F}^s})\right)\left(2^s\mathcal{F}(h_{\restriction\mathbb{F}^{n-s}})\right) \\ &= 2^{-n}\mathcal{F}(g)\mathcal{F}(h).\end{aligned}$$

□

It is immediate from Remark 12 and Theorem 13 that the following corollary holds.

Corollary 14 *For $t \in \mathbb{N}$ and $1 \le i \le t$, let $X_i \subset X = \{x_1, \ldots, x_n\}$, with $|X_i| = n_i$, be such that all X_i are pairwise disjoint. If $f(X) = \sum_{i=1}^t f_i(X_i)$, with $f_i \in B_{n_i}$, then $\mathcal{F}(f) = 2^{n-r}\prod_{i=1}^t \mathcal{F}(f_{i\restriction\mathbb{F}^{n_i}})$, with $r = n_1 + \cdots + n_t$.*

Proposition 15 *Let $f \in B_n$ be such that $f \sim_A g(x_1, \ldots, x_s) + h(x_{s+1}, \ldots, x_n)$, with $s < n$. Then*

$$\begin{aligned}\mathrm{w}(f) &= 2^{n-s}\mathrm{w}(g_{\restriction\mathbb{F}^s}) + 2^s\mathrm{w}(h_{\restriction\mathbb{F}^{n-s}}) - 2\mathrm{w}(g_{\restriction\mathbb{F}^s})\mathrm{w}(h_{\restriction\mathbb{F}^{n-s}}) \\ &= \mathrm{w}(g) + \mathrm{w}(h) - 2^{1-n}\mathrm{w}(g)\mathrm{w}(h).\end{aligned}$$

Proof We have

$$\begin{aligned}\mathrm{w}(f) &= 2^{n-1} - \frac{1}{2}\mathcal{F}(f) = 2^{n-1} - \frac{1}{2}\left(\mathcal{F}(g_{\restriction\mathbb{F}^s})\mathcal{F}(h_{\restriction\mathbb{F}^{n-s}})\right) \\ &= 2^{n-1} - \frac{1}{2}\left[\left(2^s - 2\mathrm{w}(g_{\restriction\mathbb{F}^s})\right)\left(2^{n-s} - 2\mathrm{w}(h_{\restriction\mathbb{F}^{n-s}})\right)\right] \\ &= 2^{n-s}\mathrm{w}(g_{\restriction\mathbb{F}^s}) + 2^s\mathrm{w}(h_{\restriction\mathbb{F}^{n-s}}) - 2\mathrm{w}(g_{\restriction\mathbb{F}^s})\mathrm{w}(h_{\restriction\mathbb{F}^{n-s}}) \\ &= \mathrm{w}(g) + \mathrm{w}(h) - 2^{1-n}\mathrm{w}(g)\mathrm{w}(h).\end{aligned}$$

□

We consider the balancedness of splitting functions.

Corollary 16 *Let $f \in B_n$ be such that $f \sim_A g(x_1, \ldots, x_s) + h(x_{s+1}, \ldots, x_n)$, with $s < n$. Then f is balanced if and only if at least one among g and h is balanced.*

Proof We have that f is balanced if and only if $\mathcal{F}(f) = 0$. From Theorem 13, this is equivalent to having $\mathcal{F}(g_{\restriction\mathbb{F}^s}) = 0$ or $\mathcal{F}(h_{\restriction\mathbb{F}^{n-s}}) = 0$, corresponding to having either g or h balanced. □

Remark 17 Note that from Corollary 16 we can deduce Proposition 10.

In the following proposition we consider splitting functions of a special form. This allows us to compute the exact value of their weights.

Proposition 18 *Let $f \in B_n$, with $\deg(f) = m$, be such that*

$$f \sim_A \sum_{i=0}^{k-1} \prod_{j=1}^{m} x_{mi+j},$$

where $k \in \mathbb{N}$ is such that $mk \leq n$. Then

$$\mathcal{F}(f) = 2^{n-mk}(2^m - 2)^k \quad \text{and} \quad \mathrm{w}(f) = 2^{n-1} - 2^{n-mk-1}(2^m - 2)^k.$$

Proof First, let $f_i = \prod_{j=1}^{m} x_{mi+j}$ so that $f \sim_A \sum_{i=0}^{k-1} f_i$. Then, by Corollary 14, we have $\mathcal{F}(f) = 2^{n-mk} \prod_{i=0}^{k-1} \mathcal{F}(f_{i\restriction \mathbb{F}^m})$. For all $x \in \mathbb{F}^m \setminus \{1_m\}$, observe that $f_{i\restriction \mathbb{F}^m}(x) = 0$, and $f_{i\restriction \mathbb{F}^m}(1_m) = 1$, so $\mathcal{F}(f_{i\restriction \mathbb{F}^m}) = 2^m - 2$. Thus $\mathcal{F}(f) = 2^{n-mk}(2^m - 2)^k$. Hence, we have $\mathrm{w}(f) = 2^{n-1} - \frac{1}{2}\mathcal{F}(f) = 2^{n-1} - \frac{1}{2}[2^{n-mk}(2^m - 2)^k] = 2^{n-1} - 2^{n-mk-1}(2^m - 2)^k$. □

Observe that the function f in Proposition 18 is balanced if and only if $m = 1$, that is, f is balanced if and only if it is a linear function.

Remark 19 All quadratic Bf's are splitting functions (deduced from Theorem 6) and those which are not balanced are of the form given in Proposition 18, or complements thereof, with $m = 2$. So applying Proposition 18, we obtain $\mathrm{w}(f) = 2^{n-1} - 2^{n-k-1}$ and $\mathrm{w}(f + 1) = 2^{n-1} + 2^{n-k-1}$. This result on the weight of quadratic Boolean functions is well known.

3.1 The weight computation of twisted products

In this subsection, we study the weight and balancedness of the twisted products of Bf's. We show how the weight of a Bf on n variables can be related to the weights of some other functions of lower dimension. For ease of notation, in this subsection we consider Boolean functions on $n + 1$ and $n + m$ variables, for $n, m \in \mathbb{N}$.

Any Bf in $n + 1$ variables can be expressed in the form

$$f \sim_A x_{n+1} g'(x_1, \ldots, x_n) + h'(x_1, \ldots, x_n), \tag{3.1}$$

for $g', h' \in B_n$. Observe that $x_{n+1}g'(x_1, \ldots, x_n) + h'(x_1, \ldots, x_n) = x_{n+1}(g' + h') + (1 + x_{n+1})h'$. So any Bf f in $n + 1$ variables can be written in the form

$$f \sim_A x_{n+1} g(x_1, \ldots, x_n) + (1 + x_{n+1})h(x_1, \ldots, x_n), \tag{3.2}$$

for $g, h \in B_n$. Given (3.2), we say that f is the *twisted product* of g and h. Observe that the twisted product is a special case of the form defined by

$$f \sim_A \left(\prod_{j=1}^{m} x_j\right) g(x_{m+1}, \ldots, x_{m+n}) + \left(1 + \prod_{j=1}^{m} x_j\right) h(x_{m+1}, \ldots, x_{m+n}), \tag{3.3}$$

for some positive integers m, n and Bf's g and h depending on n variables.

Remark 20 Notice that, for any Bf f, there exists a positive integer m such that f can be expressed in the form (3.3). In fact, this statement is always satisfied with $m = 1$.

Next we show that if we know the weights of g and h, then we can obtain the weight of f.

Theorem 21 *For two positive integers m, n, let $f \in B_{m+n}$ be a Bf of the form (3.3). Then*

(a) $\mathrm{w}(f) = (2^m - 1)\mathrm{w}(h_{\restriction \mathbb{F}^n}) + \mathrm{w}(g_{\restriction \mathbb{F}^n})$,
(b) *f is balanced if and only if $\mathcal{F}(g_{\restriction \mathbb{F}^n}) = -\mathcal{F}(h_{\restriction \mathbb{F}^n}) \cdot (2^m - 1)$,*
(c) *f is balanced if both g and h are balanced,*
(d) *f is not balanced if one of g and h is balanced and the other is not.*

Proof Consider f as in (3.3).

(a) Any element $X \in \mathbb{F}^{m+n}$ can be written as $X = (x, y)$ for $x \in \mathbb{F}^m$ and $y \in \mathbb{F}^n$. Using this decomposition, we have

$$\begin{aligned}\mathcal{F}(f) &= \sum_{X \in \mathbb{F}^{m+n}} (-1)^{f(X)} = \sum_{(x,y) \in \mathbb{F}^m \setminus \{1_m\} \times \mathbb{F}^n} (-1)^{h(y)} + \sum_{(x,y) \in \{1_m\} \times \mathbb{F}^n} (-1)^{g(y)} \\ &= (2^m - 1) \sum_{y \in \mathbb{F}^n} (-1)^{h(y)} + \sum_{y \in \mathbb{F}^n} (-1)^{g(y)} = (2^m - 1)\mathcal{F}(h_{\restriction \mathbb{F}^n}) + \mathcal{F}(g_{\restriction \mathbb{F}^n}).\end{aligned} \tag{3.4}$$

Since $\mathcal{F}(f) = 2^{m+n} - 2\mathrm{w}(f)$, we have

$$\begin{aligned}\mathrm{w}(f) &= 2^{n+m-1} - \frac{1}{2}\mathcal{F}(f) = 2^{n+m-1} - \frac{1}{2}\left[(2^m - 1)\mathcal{F}(h_{\restriction \mathbb{F}^n}) + \mathcal{F}(g_{\restriction \mathbb{F}^n})\right] \\ &= 2^{n+m-1} - \frac{1}{2}\left[(2^m - 1)(2^n - 2\mathrm{w}(h_{\restriction \mathbb{F}^n})) + (2^n - 2\mathrm{w}(g_{\restriction \mathbb{F}^n}))\right] \\ &= 2^{n+m-1} - \frac{1}{2}\left[2^{n+m} - 2^{m+1}\mathrm{w}(h_{\restriction \mathbb{F}^n}) + 2\mathrm{w}(h_{\restriction \mathbb{F}^n}) - 2\mathrm{w}(g_{\restriction \mathbb{F}^n})\right] \\ &= (2^m - 1)\mathrm{w}(h_{\restriction \mathbb{F}^n}) + \mathrm{w}(g_{\restriction \mathbb{F}^n}).\end{aligned}$$

(b) Recall that f is balanced if and only if $\mathcal{F}(f) = 0$. This is satisfied if and only if $(2^m - 1)\mathcal{F}(h_{\restriction \mathbb{F}^n}) + \mathcal{F}(g_{\restriction \mathbb{F}^n}) = 0$, which is equivalent to $\mathcal{F}(h_{\restriction \mathbb{F}^n}) = -\mathcal{F}(g_{\restriction \mathbb{F}^n})/(2^m - 1)$.
(c) Suppose g and h are both balanced. Then $\mathcal{F}(g_{\restriction \mathbb{F}^n}) = \mathcal{F}(h_{\restriction \mathbb{F}^n}) = 0$. From Eq. (3.4), we have that $\mathcal{F}(f) = 0$, and so f is balanced.
(d) Without loss of generality, suppose that g is balanced while h not. Then $\mathcal{F}(g_{\restriction \mathbb{F}^n}) = 0$ and $\mathcal{F}(h_{\restriction \mathbb{F}^n}) \neq 0$ which, by Eq. (3.4), implies that $\mathcal{F}(f) \neq 0$, and so f is not balanced.

□

Remark 22 Note that if f from Theorem 21 is balanced, then $(2^m - 1) \mid \mathcal{F}(g_{\restriction \mathbb{F}^n})$.

Remark 23 If $m = 1$ in Theorem 21, so that, $f = (x_{n+1})g + (1 + x_{n+1})h$, we have $\mathrm{w}(f) = \mathrm{w}(h_{\restriction \mathbb{F}^n}) + \mathrm{w}(g_{\restriction \mathbb{F}^n})$. Moreover, observe that if g and h are such that $g = h \circ \varphi + 1$, for some affine automorphism φ over $\mathbb{F}^n$, then f is balanced. Indeed, $\mathrm{w}(f) = \mathrm{w}(h_{\restriction \mathbb{F}^n}) + \mathrm{w}(g_{\restriction \mathbb{F}^n}) = \mathrm{w}(h_{\restriction \mathbb{F}^n}) + 2^n - \mathrm{w}(h \circ \varphi_{\restriction \mathbb{F}^n}) = \mathrm{w}(h_{\restriction \mathbb{F}^n}) + 2^n - \mathrm{w}(h_{\restriction \mathbb{F}^n}) = 2^n$.

Finally, we consider the weight of cubic Bf's. Generally, it is difficult to determine the weight for Bf's of degree greater than 2 (problem addressed for example in [3]). In the following proposition, we present a result which completely describes the weight of a special class of cubic functions. Using Theorem 6 and Remarks 19 and 23, the proof of Proposition 24 is a direct case-by-case computation. For this reason, we omit the proof.

Proposition 24 *Let* $f = x_{n+1}g(x_1, \ldots, x_n) + (1 + x_{n+1})h(x_1, \ldots, x_n) \in B_{n+1}$ *be cubic and such that* $\deg(g), \deg(h) \leq 2$. *Set* $q(x) = x_1x_2 + \cdots + x_{2k-1}x_{2k}$, $\bar{q}(x) = q(x) + 1$, $r(x) = x_1x_2 + \cdots + x_{2\ell-1}x_{2\ell}$ *and* $\bar{r}(x) = r(x) + 1$, *for some* $k, \ell \leq \lfloor \frac{n}{2} \rfloor$. *If* h *or* g *is balanced, then* $h \sim_A q + x_{2k+1}$ *or* $g \sim_A r + x_{2\ell+1}$ *respectively. If* h *is not balanced, then* $h \sim_A q$ *or* $h \sim_A \bar{q}$; *and if* g *is not balanced then* $g \sim_A r$ *or* $g \sim_A \bar{r}$. *Moreover,*

$$\mathrm{w}(f) = \begin{cases} 2^n, & \text{if both } h \text{ and } g \text{ are balanced,} \\ 2^{n-1}, & \text{if } h \text{ (resp. } g\text{) is bal. quad. and } g \text{ (resp. } h\text{)} = 0, \\ 2^n + 2^{n-1}, & \text{if } h \text{ (resp. } g\text{) is bal. quad. and } g \text{ (resp. } h\text{)} = 1, \\ 2^{n-1} \pm 2^{n-k-1}, & \text{if } h \text{ is not bal. quad. and } g = 0, \\ 2^n + 2^{n-1} \pm 2^{n-k-1}, & \text{if } h \text{ is not bal. quad. and } g = 1, \\ 2^{n-1} \pm 2^{n-\ell-1}, & \text{if } h = 0 \text{ and } g \text{ is not bal. quad.,} \\ 2^n + 2^{n-1} \pm 2^{n-\ell-1}, & \text{if } h = 1 \text{ and } g \text{ is not bal. quad.,} \\ 2^n \pm 2^{n-k-1}, & \text{if } h \text{ is not bal. quad. and } g \text{ is bal.,} \\ 2^n \pm 2^{n-\ell-1} & \text{if } h \text{ is bal. and } g \text{ is not bal. quad.,} \\ 2^n - 2^{n-k-1} - 2^{n-\ell-1}, & \text{if } h \sim_A q \text{ and } g \sim_A r, \\ 2^n + 2^{n-k-1} + 2^{n-\ell-1}, & \text{if } h \sim_A \bar{q} \text{ and } g \sim_A \bar{r}, \\ 2^n + 2^{n-k-1} - 2^{n-\ell-1}, & \text{if } h \sim_A \bar{q} \text{ and } g \sim_A r, \\ 2^n - 2^{n-k-1} + 2^{n-\ell-1}, & \text{if } h \sim_A q \text{ and } g \sim_A \bar{r}. \end{cases}$$

Thanks to Proposition 24, we deduce the following corollary which gives a description of all balanced cubic functions of the class $f = x_{n+1}g(x_1, \ldots, x_n) + (1 + x_{n+1})h(x_1, \ldots, x_n)$, with $\deg(g), \deg(h) \leq 2$.

Corollary 25 *With the same notation as in Proposition 24, a cubic Bf* f *is balanced if and only if one of the following holds*:

(a) *both* g *and* h *are balanced,*
(b) $g \sim_A q$ *and* $h \sim_A \bar{q}$,
(c) $g \sim_A \bar{q}$ *and* $h \sim_A q$.

Corollary 25 can be restated as follows.

Corollary 26 *Let* $f = x_{n+1}g(x_1, \dots, x_n) + (1 + x_{n+1})h(x_1, \dots, x_n)$, *with* $\deg(h), \deg(g) \leq 2$, *be a cubic Boolean function. Then f is balanced if and only if either both g and h are balanced or* $g = h \circ \varphi + 1$, *for some affine automorphism* φ.

3.2 The weight of cubic Boolean functions

Finally, we consider generic cubic Bf's, not only those that can be expressed as in Proposition 24. In this last part we go back to considering Bf's on n variables. When a Bf $f \in B_n$ is expressed in the form (3.1), that is, $f = x_1 g(x_2, \dots, x_n) + h(x_2, \dots, x_n)$, we have

$$\mathrm{w}(f) = \mathrm{w}((g + h)_{\restriction \mathbb{F}^{n-1}}) + \mathrm{w}(h_{\restriction \mathbb{F}^{n-1}}). \tag{3.5}$$

Therefore, to determine the weight of f, we can simply compute the weight of $(g + h)_{\restriction \mathbb{F}^{n-1}}$ and of $h_{\restriction \mathbb{F}^{n-1}}$. Since we are considering cubic functions, then g is quadratic and h can be affine, quadratic or cubic. If h is affine or quadratic, then $\deg(g + h), \deg(h) \leq 2$ and the weight of f is already described in Proposition 24. On the other hand, if h is cubic, then $g + h$ is also cubic and Proposition 24 cannot be directly applied. However, we can recursively repeat the process of decomposing f so that its weight is computed as the sum of the weights of some affine or quadratic functions. We now show how this can be done. Set $f = x_1 g_1(x_2, \dots, x_n) + h_1(x_2, \dots, x_n)$ with $\deg(h_1) = 3$. Then, we can write h_1 as $h_1 = x_2 g_2(x_3, \dots, x_n) + h_2(x_3, \dots, x_n)$, where $\deg(g_2) \leq 2$ and $\deg(h_2) \leq 3$. We do the same for g_1, hence $g_1 = x_2 g_2'(x_3, \dots, x_n) + h_2'(x_3, \dots, x_n)$, where $\deg(g_2') \leq 1$ and $\deg(h_2') \leq 2$. Therefore, $g_1 + h_1 = x_2(g_2 + g_2') + h_2 + h_2'$. Notice that the cubic terms of h_1 coincide with those of $g_1 + h_1$. So

$$\begin{aligned}
&\mathrm{w}(h_{1\restriction \mathbb{F}^{n-1}}) = \mathrm{w}((g_2 + h_2)_{\restriction \mathbb{F}^{n-2}}) + \mathrm{w}(h_{2\restriction \mathbb{F}^{n-2}}),\\
&\mathrm{w}((g_1 + h_1)_{\restriction \mathbb{F}^{n-1}}) = \mathrm{w}((g_2 + h_2 + g_2' + h_2')_{\restriction \mathbb{F}^{n-2}}) + \mathrm{w}((h_2 + h_2')_{\restriction \mathbb{F}^{n-2}}).
\end{aligned}$$

If the degree of h_2 is smaller than 3, the weights of $(g_1 + h_1)_{\restriction \mathbb{F}^{n-1}}$ and of $h_{1\restriction \mathbb{F}^{n-1}}$ are described in Proposition 24. Otherwise, we continue with the decomposition of the functions, namely h_2, g_2, h_2', g_2', e.g. $h_2 = x_3 g_3(\dots) + h_3(\dots)$, and we recursively apply the same approach. Every decomposition step doubles the number of functions for which we want to compute the weight. Notice that the variables that determine the decomposition of the Bf, namely $x_1, x_2, \dots$ in the computations above, should be selected in such a way as to minimize the number of recursive steps needed. Our choice is to select the variable which occurs most frequently in $\mathrm{Term}_3(h_i)$, where $\mathrm{Term}_3(h_i)$ is the set of all cubic terms of h_i. Other choice strategies are possible and we do not claim that our strategy is optimal. We leave it as an open problem. To simplify the description of the procedure, we call *quadratic representation* the final decomposition of the involved cubic Boolean functions.

We use this idea to build an algorithm that computes the weight of any cubic Bf.

Algorithm 1 Computing the weight of a cubic Boolean function f on n variables:

Table 1 Average value t of the number of decompositions needed to reduce a randomly-generated cubic Bf in B_n into a quadratic representation

n	t	n	t	n	t	n	t
5	$n-4$	30	$n-6$	55	$n-7$	80	$n-7$
10	$n-5$	35	$n-6$	60	$n-7$	85	$n-7$
15	$n-6$	40	$n-7$	65	$n-7$	90	$n-7$
20	$n-6$	45	$n-7$	70	$n-7$	95	$n-7$
25	$n-6$	50	$n-7$	75	$n-7$	100	$n-8$

Table 2 Average value t of the number of decompositions needed to reduce a randomly-generated cubic Bf in B_n (with at most n^2 cubic terms) into a quadratic representation

n	t	3/4n	n	t	3/4n	n	t	3/4n	n	t	3/4n
5	2	3	30	19	22	55	40	41	80	61	60
10	5	7	35	23	26	60	43	45	85	65	63
15	8	11	40	27	30	65	48	48	90	70	67
20	12	15	45	31	33	70	50	52	95	74	71
25	16	18	50	35	37	75	57	56	100	77	75

Input: (the ANF of) a cubic function $f \in B_n$,
Output: $\mathrm{w}(f)$,
Step 1: express f in the form $f = x_i g(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) + h(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ where x_i occurs most frequently in $\mathrm{Term}_3(f)$,
Step 2: if $\deg(h) \leq 2$, compute $\mathrm{w}(g+h)$ and $\mathrm{w}(h)$ and apply (3.5) to return $\mathrm{w}(f)$,
Step 3: otherwise, recursively compute the weights of $g+h$ and h by applying **Step 1** and **Step 2**

We now briefly discuss the complexity of determining the weight of a cubic Boolean function using the above algorithm. Suppose that we are able to obtain a function h_t of degree at most 2 after t iterations of the decomposition. Therefore, we end up with 2^t functions of degree at most 2, defined over $\mathbb{F}^{n-t}$, whose weight we have to compute. We performed some computational experiments to estimate the value of t. In Table 1 we report the average value of t for 200 randomly-generated cubic Boolean functions over $\mathbb{F}^n$, with n varying from 5 to 100. Since we are only interested in the cubic terms of a cubic Bf, to generate it we selected randomly an integer r between 1 and the number of possible cubic terms, and then we selected r cubic terms at random. The decompositions were performed using the strategy described in Algorithm 1.

The values reported in Table 1 are quite accurate. For example, in the case $n = 100$ the minimum value obtained is 44, the maximum 96, the mean is 92.115 and the variance is 39.222.

In Table 2 we report the results obtained when considering 200 randomly-selected cubic Boolean functions with more sparse terms, that is, $f \in B_n$ cubic with at most n^2 cubic terms. The functions are generated as described previously, with the number of possible cubic terms restricted by $1 \leq r \leq n^2$. As before, the decompositions were performed using the strategy described in Algorithm 1. We show in the

table that, for the values of n considered, the average number of decompositions needed is around $\frac{3}{4}n$.

Similarly to the previous table, the values reported in Table 2 are quite accurate. For example, in the case $n = 100$ the minimum value obtained is 15, the maximum 87, the mean is 77.25 and the variance is 151.4275. We leave it as an open problem to investigate the effectiveness of our method for Boolean functions with other restrictions on Term_3.

We now study the complexity of determining the weight of a quadratic Bf.

(a) We consider the following operations to be elementary: bit addition, bit multiplication, variable multiplications (e.g. $x_1 \cdot x_2 \to x_1x_2$). The goal of our final estimate will be in terms of elementary operations with the big O notation.

(b) We recall that given a quadratic $g \in B_k$, we can obtain an affine equivalent map as in Theorem 6. This result is well known, see for instance [4], Subsection 5.2.1. Indeed, if x_1x_2 is a term of g, then $g = x_1x_2 + x_1g_1(x_3, \dots, x_k) + x_2g_2(x_3, \dots, x_k) + g_{1,2}(x_3, \dots, x_k)$, for some $g_1, g_2, g_{1,2}$ with $\deg(g_1), \deg(g_2) \le 1$ and $\deg(g_{1,2}) \le 2$. Hence $g = (x_1 + g_2)(x_2 + g_1) + g_1g_2 + g_{1,2}$ is affine equivalent to $x_1x_2 + g_1g_2 + g_{1,2}$. Applying this method recursively, in at most $k/2$ iterations we obtain an affine equivalent map as in Theorem 6, and using Remark 19 we compute the weight of g. In the following, we will use the word *(decomposition) step* to refer to one of the above iterations.

(c) Now we analyse the r-th decomposition step of (b). In this step, we consider a quadratic term in g, namely x_ix_j.

We consider the representation of g as $g = x_ix_j + x_ig_i + x_jg_j + g_{i,j}$.

Notice that, since we are at the r-th step of the decomposition, $g_i, g_j, g_{i,j}$ depend on at most $k - 2r$ variables and g_i, g_j have at most $k - 2r + 1$ terms. From the ANF of g we determine g_i, g_j and $g_{i,j}$. Indeed, $g_{i,j} = g(x_i = 0, x_j = 0)$, $g_i = g(x_i = 1, x_j = 0) + g_{i,j}$ and $g_j = g(x_i = 0, x_j = 1) + g_{i,j}$, where, with abuse of notation, $g(x_i = a, x_j = b)$ indicates that we consider the ANF of g with the substitutions $x_i = a$ and $x_j = b$. In the $(r + 1)$-th step we will consider $g \leftarrow g_ig_j + g_{i,j}$.

(d) We now evaluate the complexity of every computation in the r-th step. The cost of computing $g_{i,j}$ is bounded by the cost of performing a full evaluation $g(a_1, \dots, a_k)$, which itself is bounded by k^2 multiplications and k^2 additions. The cost for g_i, g_j is similar, but we have to add k^2 additions. So the overall cost of computing $g_i, g_j, g_{i,j}$ is bounded by the cost of performing $8k^2$ elementary operations. The multiplication of the two linear polynomials g_i and g_j costs at most k^2 variable multiplications, since they have at most $k - 2r - 1$ terms. Therefore the computation of the input of the next step (the new g) is bounded by $9k^2$ elementary operations.

(e) Hence, from (b) and (d), we conclude that the cost of determining the weight of a quadratic Bf in k variables is at most $\text{cost}_k = 9k^2 \cdot \frac{k}{2} = \frac{9}{2}k^3 \le 5k^3$ elementary operations.

(f) Therefore, by considering $k = n - t$, the complexity of computing the weight of a cubic f is at most $2^t \cdot \text{cost}_{n-t} = 2^t \cdot \frac{9}{2}(n - t)^3$ elementary operations.

(g) If we consider $t \approx \frac{3}{4}n$ (as we saw in Table 2) we have $2^{\frac{3}{4}n} cost_{n/4} \approx O(2^{\frac{3}{4}n}(n/4)^3) = O(2^{\frac{3}{4}n} n^3)$.

We now compare the complexity of Algorithm 1 with the complexity of computing the weight of a Boolean function with some other methods. For a more general overview, we also consider methods admitting distinct types of input (Algorithm 1 takes as input the ANF of a Boolean function). Recall that the truth table of $f \in B_n$ is $T_f = [f(u) : u \in \mathbb{F}^n]$. So, to compute the weight of f we can count how many 1's appear in T_f.

- If for input we have the truth table of f, the cost is at most 2^n additions.
- If for input we have some information on the Walsh transform, in particular its value in zero, the computation is immediate ($\mathcal{W}_f(0) = 2^n - 2\mathrm{w}(f)$).
- If for input we have the ANF of f, either we compute the truth table (the cost is at least 2^n evaluations), or we can compute the Walsh transform in zero (again the cost is 2^n evaluations).

Therefore, according to our estimates, given the ANF of a cubic Boolean function with a limited number of cubic terms, as described in Table 2, our algorithm is more efficient than the other known procedures described.

4 Nonlinearity of Boolean functions

Another important cryptographic property of Boolean functions is the nonlinearity. We begin this section by studying this property for a function with terms of the same degree but pairwise disjoint sets of variables, as in Proposition 18.

Proposition 27 *Let $f \in B_n$, with $\deg(f) = m > 1$, be such that*

$$f \sim_A \sum_{t=0}^{k-1} \prod_{j=1}^{m} x_{mt+j},$$

where k is the number of terms and $mk \leq n$. Then $\mathcal{N}(f) = 2^{n-1} - 2^{n-mk-1}(2^m - 2)^k$.

Proof Let $f_i = \prod_{j=1}^{m} x_{mi+j}$. Then $f \sim_A \sum_{i=0}^{k-1} f_i$. Let $l_\alpha(x) = \alpha \cdot x$, where $\alpha, x \in \mathbb{F}^n$. Observe that $f + l_\alpha$ is balanced if l_α has some variables which are not in f (see Proposition 10) and in this case, we have $\mathcal{W}_f(\alpha) = \mathcal{F}(f + l_\alpha) = 0$. Thus we can assume that $l_\alpha(x) = l_a(X) = a \cdot X$, with $a = (a_0, \ldots, a_{k-1})$ and $X = (y_0, \ldots, y_{k-1})$ in $(\mathbb{F}^m)^k$, so that all variables in l_a are also in f. By Corollary 14, we have

$$\mathcal{W}_f(\alpha) = \mathcal{F}(f + l_a) = 2^{n-mk} \prod_{i=0}^{k-1} \mathcal{F}([f_i + l_{a_i}]_{\restriction \mathbb{F}^m}).$$

Recall that $\mathcal{N}(f) = 2^{n-1} - \frac{1}{2} \max_{\alpha \in \mathbb{F}^n} |\mathcal{W}_f(\alpha)|$. Clearly, $|\mathcal{W}_f(\alpha)|$ is maximal if all $|\mathcal{F}([g_i + l_{a_i}]_{\restriction \mathbb{F}^m})|$ are maximal. Now, $\mathcal{F}([f_i + l_{a_i}]_{\restriction \mathbb{F}^m}) = 2^m - 2\mathrm{w}([f_i + l_{a_i}]_{\restriction \mathbb{F}^m})$ and it is clear that $\mathrm{w}([f_i + l_{a_i}]_{\restriction \mathbb{F}^m}) \notin \{0, 2^m\}$. So, without loss of generality, $|\mathcal{F}([f_i + l_{a_i}]_{\restriction \mathbb{F}^m})|$ is maximal if $a_i = 0_m$ since in this case $\mathrm{w}([f_i + l_{a_i}]_{\restriction \mathbb{F}^m}) = \mathrm{w}(f_{i\restriction \mathbb{F}^m}) = 1$. Thus, $|\mathcal{W}_f(\alpha)|$ is maximal if, for all i, we have $\mathcal{F}([f_i + l_{a_i}]_{\restriction \mathbb{F}^m}) = \mathcal{F}(f_{i\restriction \mathbb{F}^m}) = 2^m - 2$, implying that it is maximal when $\alpha = 0_n$. We have $\mathcal{W}_f(0_n) = 2^{n-mk}(2^m - 2)^k$ and therefore $\mathcal{N}(f) = 2^{n-1} - 2^{n-mk-1}(2^m - 2)^k$. □

Remark 28 Recall that $f \in B_n$ is bent if and only if $\mathcal{N}(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$. We deduce from Proposition 27 that f is bent if and only if $m = 2$ and $k = n/2$, for n even, otherwise $2^{n-mk-1}2^k(2^{m-1} - 1)^k$ would be equal to $2^{\frac{n}{2}-1}$, for some positive integer k, contradicting the fact that $(2^{m-1} - 1) \nmid 2^{\frac{n}{2}-1}$ for $m > 2$.

We study the nonlinearity for a Bf $f \in B_{n+m}$ of the form (3.3).

Theorem 29 *Let f be a Bf of the form (3.3), that is, $f = \left(\prod_{j=1}^m x_j\right) g(x_{m+1}, \ldots, x_{m+n}) + \left(1 + \prod_{j=1}^m x_j\right) h(x_{m+1}, \ldots, x_{m+n})$. Let $\alpha = (a, b) \in \mathbb{F}^m \times \mathbb{F}^n$, with $a = (a_1, \ldots, a_m)$ and $b = (b_1, \ldots, b_n)$. Then*

(i) $$\mathcal{W}_f(\alpha) = \begin{cases} (2^m - 1)\mathcal{W}_{h_{\restriction \mathbb{F}^n}}(b) + \mathcal{W}_{g_{\restriction \mathbb{F}^n}}(b), & \text{if } a = 0_m, \\ (-1)^{\lambda_a}\left(\mathcal{W}_{g_{\restriction \mathbb{F}^n}}(b) - \mathcal{W}_{h_{\restriction \mathbb{F}^n}}(b)\right), & \text{otherwise,} \end{cases} \quad \text{with } \lambda_a = a_1 + \cdots + a_m;$$

(ii) $$\mathcal{N}(f) \geq (2^m - 1)\mathcal{N}(h_{\restriction \mathbb{F}^n}) + \mathcal{N}(g_{\restriction \mathbb{F}^n}).$$

Proof Set $X = (y, x) \in \mathbb{F}^m \times \mathbb{F}^n$, with $y = (x_1, \ldots, x_m)$ and $x = (x_{m+1}, \ldots, x_{m+n})$. Then

$$\begin{aligned} \mathcal{W}_f(\alpha) &= \sum_{X \in \mathbb{F}^{m+n}} (-1)^{f(X)+\alpha \cdot X} \\ &= \sum_{(y,x) \in \mathbb{F}^m \setminus \{1_m\} \times \mathbb{F}^n} (-1)^{h(x)+a \cdot y+b \cdot x} + \sum_{(y,x) \in \{1_m\} \times \mathbb{F}^n} (-1)^{g(x)+a \cdot y+b \cdot x} \\ &= \sum_{(y,x) \in \mathbb{F}^m \times \mathbb{F}^n} (-1)^{h(x)+a \cdot y+b \cdot x} - \sum_{x \in \mathbb{F}^n} (-1)^{h(x)+b \cdot x+\lambda_a} + \sum_{x \in \mathbb{F}^n} (-1)^{g(x)+b \cdot x+\lambda_a} \\ &= \left(\sum_{y \in \mathbb{F}^m} (-1)^{a \cdot y}\right) \mathcal{W}_{h_{\restriction \mathbb{F}^n}}(b) - (-1)^{\lambda_a} \mathcal{W}_{h_{\restriction \mathbb{F}^n}}(b) + (-1)^{\lambda_a} \mathcal{W}_{g_{\restriction \mathbb{F}^n}}(b) \\ &= \begin{cases} (2^m - 1)\mathcal{W}_{h_{\restriction \mathbb{F}^n}}(b) + \mathcal{W}_{g_{\restriction \mathbb{F}^n}}(b), & \text{if } a = 0_m, \\ (-1)^{\lambda_a}\left[\mathcal{W}_{g_{\restriction \mathbb{F}^n}}(b) - \mathcal{W}_{h_{\restriction \mathbb{F}^n}}(b)\right], & \text{otherwise.} \end{cases} \end{aligned}$$

So we have

$$|\mathcal{W}_f(\alpha)| \leq \begin{cases} (2^m-1)|\mathcal{W}_{h_{|\mathbb{F}^n}}(b)| + |\mathcal{W}_{g_{|\mathbb{F}^n}}(b)|, & \text{if } a = 0_m, \\ |\mathcal{W}_{g_{|\mathbb{F}^n}}(b)| + |\mathcal{W}_{h_{|\mathbb{F}^n}}(b)|, & \text{otherwise.} \end{cases}$$

Therefore, for any $\alpha = (a, b) \in \mathbb{F}^m \times \mathbb{F}^n$, we have

$$|\mathcal{W}_f(\alpha)| \leq (2^m-1)|\mathcal{W}_{h_{|\mathbb{F}^n}}(b)| + |\mathcal{W}_{g_{|\mathbb{F}^n}}(b)|.$$

So

$$\begin{aligned} \mathcal{N}(f) &= 2^{n+m-1} - \frac{1}{2} \max_{\alpha \in \mathbb{F}^m \times \mathbb{F}^n} |\mathcal{W}_f(\alpha)| \\ &\geq 2^{n+m-1} - \frac{1}{2} \max_{b \in \mathbb{F}^n} \left((2^m-1)|\mathcal{W}_{h_{|\mathbb{F}^n}}(b)| + |\mathcal{W}_{g_{|\mathbb{F}^n}}(b)| \right) \\ &\geq 2^{n+m-1} - \frac{1}{2}(2^m-1) \max_{b \in \mathbb{F}^n} |\mathcal{W}_{h_{|\mathbb{F}^n}}(b)| - \frac{1}{2} \max_{b \in \mathbb{F}^n} |\mathcal{W}_{g_{|\mathbb{F}^n}}(b)| \\ &= (2^m-1)2^{n-1} - \frac{1}{2}(2^m-1) \max_{b \in \mathbb{F}^n} |\mathcal{W}_{h_{|\mathbb{F}^n}}(b)| + 2^{n-1} - \frac{1}{2} \max_{b \in \mathbb{F}^n} |\mathcal{W}_{g_{|\mathbb{F}^n}}(b)| \\ &= (2^m-1)\mathcal{N}(h_{|\mathbb{F}^n}) + \mathcal{N}(g_{|\mathbb{F}^n}). \end{aligned}$$

□

Remark 30 Since the nonlinearity is invariant under affine equivalence, the second condition in Theorem 29 is satisfied also for Boolean functions which are affine equivalent to f.

Remark 31 By Theorem 29 with $m = 1$, the nonlinearity of

$$f \sim_A x_{n+1} g(x_1, .., x_n) + (1 + x_{n+1}) h(x_1, \ldots, x_n)$$

satisfies $\mathcal{N}(f) \geq \mathcal{N}(h_{|\mathbb{F}^n}) + \mathcal{N}(g_{|\mathbb{F}^n})$.

It is immediate from Theorem 7 and Remark 31 that the following corollary holds.

Corollary 32 *Let f be as described in Proposition 24. Then*

$$\mathcal{N}(f) \geq \begin{cases} 2^{n-1} - 2^{n-k-1}, & \text{if } g \text{ is quadratic and } h \text{ affine,} \\ 2^{n-1} - 2^{n-\ell-1}, & \text{if } g \text{ is affine and } h \text{ quadratic,} \\ 2^n - 2^{n-k-1} - 2^{n-\ell-1}, & \text{if both } g \text{ and } h \text{ are quadratic.} \end{cases}$$

Corollary 32 suggests a way of constructing Bf's with high non-linearity. For example, following the same notation as in the corollary above with $n = 2r + 1$, if we consider two quadratic Bf's in B_n with $k = \ell = \frac{n-1}{2} = r$, then the non-linearity of $f \in B_{n+1}$ is at least $2^n - 2^{n-r} = 2^n - 2^{\frac{n+1}{2}}$.

5 Conclusion

This paper focuses on Boolean functions of particular forms. We studied the class of splitting functions and a special class of cubic functions, determining their weight, balancedness and nonlinearity. We also studied the weight and nonlinearity of a generic Boolean function by means of the weights and nonlinearities of its "decomposition" functions, defined on lower dimensions. We provide a procedure for computing the weight of a generic cubic Boolean function. We performed some computational analysis on cubic Boolean functions with sparse cubic terms, and we showed that in this case our procedure is more efficient than computing it through the truth table.

Acknowledgements The results in this paper appear partially in the last author's MSc thesis and mostly in the first author's PhD thesis, both supervised by the second author. The authors want to thank the anonymous reviewers for the useful comments that strongly contributed to the improvement of this work.

Funding Open access funding provided by Università degli Studi di Trento within the CRUI-CARE Agreement.

References

1. Beth, T., Ding, C.: On almost perfect nonlinear permutations. In: Advances in Cryptology - EUROCRYPT '93. vol 765, pp. 65–76. Springer, Berlin, Heidelberg (1993)
2. Braeken, A., Borissov, Y., Nikova, S., Preneel, B.: Classification of cubic $(n-4)$-resilient Boolean functions. IEEE Trans. Inf. Theory **52**(4), 1670–1676 (2006)
3. Carlet C.: A transformation on Boolean functions, its consequences on some problems related to Reed-Muller codes. In: Cohen G., Charpin P. (eds.) Adv. in crypt.-Eurocrypt'90. LNCS, vol 473, pp 42-50. Springer, Berlin, Heidelberg (1991)
4. Carlet, C.: Boolean Functions for Cryptography and Coding Theory. Cambridge University Press, Cambridge (2021)
5. Chee, S., Lee, S., Kim K.: Semi-bent Functions. In: Pieprzyk, J., Safavi-Naini, R. (eds.) Advances in Cryptology-ASIACRYPT'94. In: Proceedings 4th International Conference on the Theory and Applications of Cryptology, vol 917, pp 107-118. Springer, Wollongong.(1994)
6. Cusick, T. W., Stanica, P.: Chapter 6 - Special Types of Boolean Functions, Editor(s): Thomas W. Cusick, Pantelimon Stanica, Cryptographic Boolean Functions and Applications (Second Edition), Academic Press, Pages 109-142 (2017)
7. MacWilliams, F.-J., Sloane, N.-J.-A.: The Theory of Error-Correcting Codes. Elsevier, New York (1977)
8. Tang, D., Zhang, W., Tang, X.: Construction of balanced Boolean functions with high nonlinearity and good autocorrelation properties. Des. Codes Cryptogr. **60**, 77–91 (2010)
9. Wu, C., Feng, D.: Boolean Functions and Their Applications in Cryptography. Springer, New York (2016)