2022-06

C-DIFFERENTIALS AND GENERALIZED CRYPTOGRAPHIC PROPERTIES OF VECTORIAL BOOLEAN AND P-ARY FUNCTIONS

Geary, Aaron C.

Monterey, CA; Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

DISSERTATION

c-DIFFERENTIALS AND GENERALIZED CRYPTOGRAPHIC PROPERTIES OF VECTORIAL BOOLEAN AND p-ARY FUNCTIONS

by

Aaron C. Geary

June 2022

Dissertation Supervisor: Pantelimon Stănică

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2022	**3. REPORT TYPE AND DATES COVERED** Dissertation

4. TITLE AND SUBTITLE
c-DIFFERENTIALS AND GENERALIZED CRYPTOGRAPHIC PROPERTIES OF VECTORIAL BOOLEAN AND *p*-ARY FUNCTIONS

5. FUNDING NUMBERS

6. AUTHOR(S) Aaron C. Geary

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

This dissertation investigates a newly defined cryptographic differential, called a *c*-differential, and its relevance to the nonlinear substitution boxes of modern symmetric block ciphers. We generalize the notions of perfect nonlinearity, bentness, and avalanche characteristics of vectorial Boolean and *p*-ary functions using the *c*-derivative and a new autocorrelation function, while capturing the original definitions as special cases (i.e., when c=1). We investigate the *c*-differential uniformity property of the inverse function over finite fields under several extended affine transformations. We demonstrate that *c*-differential properties do not hold in general across equivalence classes typically used in Boolean function analysis, and in some cases change significantly under slight perturbations. Thus, choosing certain affine equivalent functions that are easy to implement in hardware or software without checking their *c*-differential properties could potentially expose an encryption scheme to risk if a *c*-differential attack method is ever realized. We also extend the *c*-derivative and *c*-differential uniformity into higher order, investigate some of their properties, and analyze the behavior of the inverse function's second order *c*-differential uniformity. Finally, we analyze the substitution boxes of some recognizable ciphers along with certain extended affine equivalent variations and document their performance under *c*-differential uniformity.

14. SUBJECT TERMS
cryptography, vectorial Boolean functions, *p*-ary functions, substitution boxes, nonlinearity, bent functions, avalanche characteristics, differential cryptanalysis, differential uniformity, higher order discrete derivatives, communications security

15. NUMBER OF PAGES
157

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

c-DIFFERENTIALS AND GENERALIZED CRYPTOGRAPHIC PROPERTIES OF VECTORIAL BOOLEAN AND *p*-ARY FUNCTIONS

Aaron C. Geary
Commander, United States Navy
BS, United States Naval Academy, 2003
MS, Applied Mathematics, Naval Postgraduate School, 2009

Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY IN APPLIED MATHEMATICS

from the

NAVAL POSTGRADUATE SCHOOL
June 2022

Approved by:

Pantelimon Stănică
Department of
Applied Mathematics
Dissertation Supervisor
Dissertation Chair

George W. Dinolt
Department of
Computer Science

Ralucca Gera
Department of Applied
Mathematics

Thor Martinsen
Department of
Applied Mathematics

Luis Medina
University of Puerto
Rico, Rio Piedras

Approved by:

Francis X. Giraldo
Chair, Department of Applied Mathematics

Michael E. Freeman
Vice Provost of Academic Affairs

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This dissertation investigates a newly defined cryptographic differential, called a *c*-differential, and its relevance to the nonlinear substitution boxes of modern symmetric block ciphers. We generalize the notions of perfect nonlinearity, bentness, and avalanche characteristics of vectorial Boolean and *p*-ary functions using the *c*-derivative and a new autocorrelation function, while capturing the original definitions as special cases (i.e., when $c=1$). We investigate the *c*-differential uniformity property of the inverse function over finite fields under several extended affine transformations. We demonstrate that *c*-differential properties do not hold in general across equivalence classes typically used in Boolean function analysis, and in some cases change significantly under slight perturbations. Thus, choosing certain affine equivalent functions that are easy to implement in hardware or software without checking their *c*-differential properties could potentially expose an encryption scheme to risk if a *c*-differential attack method is ever realized. We also extend the *c*-derivative and *c*-differential uniformity into higher order, investigate some of their properties, and analyze the behavior of the inverse function's second order *c*-differential uniformity. Finally, we analyze the substitution boxes of some recognizable ciphers along with certain extended affine equivalent variations and document their performance under *c*-differential uniformity.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

AES	Advanced Encryption Standard
ANF	algebraic normal form
APcN	almost perfect c-nonlinear
APN	almost perfect nonlinear
BF	Boolean function
cDU	c-differential uniformity
CCZ	Carlet-Charpin-Zinoviev
DES	Data Encryption Standard
DDT	difference distribution table
DU	differential uniformity
EA	extended affine
GAC	Global Avalanche Characteristics
KDSB	key dependent S-box
PcN	perfect c-nonlinear
PC	propagation criterion
PN	perfect nonlinear
SAC	strict avalanche criterion
S-box	substitution box
SPN	substitution permutation network

WHT Walsh-Hadamard transform

xor exclusive or

Executive Summary

Differential cryptanalysis encompasses a group of techniques aimed at exploiting how certain pairs of differences, called differentials, trace through an encryption scheme in an attempt to recover information about the secret key. Recently, researchers introduced a new type of differential, called a c-differential, that has the potential to expand differential cryptanalysis against block ciphers in a new direction. In this dissertation, we further the theoretical and practical investigation of this c-differential and its potential impact on the substitution boxes of modern symmetric block ciphers.

This new c-differential immediately leads to the notion of a discrete c-derivative. Many essential cryptographic properties of vectorial Boolean or p-ary functions can be described and characterized by the behavior of a function's derivatives. Equipped with the c-derivative, we introduce a new autocorrelation function, and with this tool we generalize multiple cryptographic properties of substitution boxes. These generalizations include the notions of perfect nonlinearity, bentness, and avalanche characteristics. Naturally, the traditional definitions of these properties are captured as special cases of our generalizations, specifically when $c = 1$. We also extend the c-derivative and c-differential into higher order, investigate some of their properties, and analyze the behavior of the inverse function's second order c-differential uniformity.

We further investigate and quantify the resistance of certain vectorial Boolean and p-ary functions against this new potential differential attack. Using the c-differential uniformity property that was introduced along with the c-differential, we study the inverse function and some extended affine transformations over finite fields. We also analyze real-world substitution boxes of known ciphers to determine their c-differential properties. We demonstrate that small perturbations to functions highly resistant to classical differential attacks can change their resistance to this new type of attack, sometimes significantly. This is important because functions often maintain essential nonlinearity properties through certain equivalence relations. Cryptographic designers have the option to choose a function that is equivalent to one with good properties but that is easier to implement on specific hardware or software. This may no longer be a safe option under the threat of a c-differential attack.

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgments

Profound gratitude goes to my dissertation committee for their guidance and instruction, chief among them my supervisor, Professor Pantelimon Stănică. I was fortunate enough to have him as the instructor for my first graduate course in mathematics at the Naval Postgraduate School. Ten years later, after being selected by the U.S. Navy for the permanent military professor program at the United States Naval Academy, there was no doubt who I wanted to be my PhD supervisor. I will forever be grateful to have studied under Pante. I would also like to thank CAPT Thor Martinsen, a member of my dissertation and qualifying exam committees, but also a career mentor. To the other members of my committee, Professors Ralucca Gera, Luis Medina, and George Dinolt, I offer my sincere appreciation. I move forward much stronger from their tutelage.

There are many others I owe thanks to and without whom this milestone would not have been possible. USNA Professor Emeritus Mark Kidwell, along with the rest of the faculty, provided a solid foundation of critical thinking and problem solving that have served me well. Thanks to Professors Chris Frenzen and Hong Zhou for their support in the qualification exam process. Thanks also to Bard Mansager, who convinced me to apply for a dual major program shortly after my first arrival in Monterey. Without this suggestion and his help navigating the process, it is doubtful I would have been in position to be selected for the PMP program.

My biggest thanks is reserved to those closest and most important to me, my family and friends. My parents Bob and Linda, my brother and sisters, and many close friends across the globe have been extremely supportive for which I am eternally grateful. To my wife, Sandy, son, Parker, and daughter, Alivia: you are the center of my universe–thank you from the bottom of my heart. This achievement is as much yours as it is mine.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

1.1 Background

Modern digital communications, and by extension our modern society, rely heavily upon cryptography to ensure the security and integrity of data being sent over public channels, most notably the internet. The fundamental cryptographic principles of *confusion* and *diffusion*, introduced by Shannon [1] in 1949, have remained remarkably relevant for over seven decades and continue to be foundational goals of any encryption scheme. Confusion refers to a highly complex relationship between plaintext, secret key, and ciphertext that cannot be well-approximated by simpler systems. Diffusion means that small changes to the input of an encryption or decryption algorithm should result in large changes in the corresponding ciphertext or plaintext.

In order to realize the principle of confusion in real-world systems, cryptographers often rely upon nonlinear substitution boxes, or "S-boxes." In mathematics, these S-boxes are also known as vectorial Boolean functions or (n, m)-functions, since they take as input an n-tuple of bits and output an m-tuple of bits. The proper design of secure S-boxes for symmetric block ciphers is a challenging problem. There are many competing demands—such as speed, efficiency, and security—and multiple potential vulnerabilities that must be addressed at the same time. Cryptographic designers must accept tradeoffs between competing criteria, inevitably achieving suboptimal performance in some respect.

One of the primary classes of attacks on block ciphers that S-boxes must be designed to protect against is differential cryptanalysis. A classical differential attack exploits the fact that certain differences between plaintexts result in high probability outputs into the last round of the cipher. If these probabilities are high enough, an attacker gains some statistical insight into the secret key, thus potentially compromising confidentiality. Resistance to this attack can often be quantified by a property of S-boxes called differential uniformity. Since the public introduction of differential cryptanalysis techniques in the late 1980s, there have been multiple extensions and modifications, some based on specific ciphers. Recently, a new

type of differential was introduced in [2]. This new multiplicative output differential, which is called a "c-differential," along with the related c-differential uniformity, has the potential to expand upon differential cryptanalysis and create new vulnerabilities for encryption schemes to deal with.

The goal of this research has been to investigate this new differential and determine its relevance to the substitution boxes of modern symmetric block ciphers. Specifically, we generalize existing vectorial Boolean function properties using the c-derivative to account for the threat posed by the new c-differential. Additionally, we have sought to analyze some theoretical classes and real-world applications of vectorial Boolean functions to determine their c-differential properties. As a result of this mathematical and computational analysis, our aim is to provide insight into how functions used as primitives in block cipher S-boxes can be protected against the threat of a c-differential attack.

1.2 Contributions

This dissertation makes the following scholarly contributions:

- We define new crosscorrelation and associated autocorrelation functions using the new c-derivative.
- We use this new autocorrelation function to extend the notion of perfect nonlinearity based on balanced derivatives into a property called perfect c-nonlinearity.
- We extend the notion of vectorial Boolean function bentness into what we call c-differential bentness by using a Walsh-Hadamard transform product motivated by the p-ary definition of a bent function.
- We characterize 0-differential bent functions and provide examples of perfect c-nonlinear functions for $c \neq 1$, even in characteristic 2.
- We generalize avalanche characteristics—specifically the strict avalanche criterion, propagation criterion, and the global avalanche characteristics—using the c-derivative and capture bounds on these properties.
- We demonstrate how small perturbations of the multiplicative inverse function, namely the extended affine transformation of adding certain linearized monomials or polynomials, can result in a significant change in the c-differential uniformity.
- We summarize and organize the current findings on c-differential uniformity from

multiple researchers into one location.

- We extend the first order c-derivative into higher order c-derivatives, capture properties of these extensions, and investigate higher order c-differential uniformity of the inverse function.
- We conduct an analysis of many real-world substitution boxes, computing their c-differential uniformities and documenting how certain extended affine transformations change their c-differential uniformity.

1.3 Dissertation Organization

This dissertation is divided into seven chapters. In addition to this introductory chapter, the remaining chapters are laid out as follows. Chapter 2 contains preliminary definitions, basic material on Boolean, vectorial Boolean, and p-ary functions with a focus on cryptographic properties along with multiple examples and a description of differential cryptanalysis. Chapter 3 introduces the c-differential, c-derivative, and generalizes many existing vectorial Boolean function properties using the c-derivative. Chapter 4 is an extensive look into the c-differential uniformity of the multiplicative inverse function over finite fields under extended affine equivalence, a popular S-box primitive. The c-derivative and corresponding c-differential uniformity are extended into higher order and investigated in Chapter 5, and Chapter 6 is our analysis of real-world S-boxes with results from a c-differential perspective. Finally, Chapter 7 concludes the dissertation and contains some follow-on research possibilities.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Cryptographic Properties of Boolean and Vectorial Boolean Functions

This chapter establishes the foundation upon which the rest of this dissertation will depend. We introduce notations, definitions, properties, and background information which we will use in the following chapters. We provide multiple examples in order to demonstrate certain properties and prepare for expanding these notions in the rest of the dissertation.

2.1 Preliminaries

We denote the set of integers, real numbers, and complex numbers by $\mathbb{Z}$, $\mathbb{R}$, and $\mathbb{C}$, respectively. The complex conjugate of an element z will be written as $\bar{z}$. For a positive integer n and p a prime number, we let $\mathbb{F}_{p^n}$ be the finite field with p^n elements, and $\mathbb{F}_{p^n}^* = \mathbb{F}_{p^n} \setminus \{0\}$. That is, $\mathbb{F}_{p^n}^*$ is $\mathbb{F}_{p^n}$ without the additive identity (0) element. For $a \neq 0$, we often write $\frac{1}{a}$ to mean the inverse of a in the multiplicative group of the finite field under discussion. $\mathbb{Z}_n$ will be used for the ring of integers modulo n.

We let $\mathbb{F}_p^n$ be the n-dimensional vector space over $\mathbb{F}_p$. We use **bold font** for vectors. The elements of this vector space are n-tuples $\mathbf{x}$, such that $\mathbf{x} = \{x_1, x_2, x_3, \ldots, x_n\}$ with $x_i \in \mathbb{F}_p$. When there is no danger of confusion, we may write the vectors without commas. In the case of $p = 2$, we say $\mathbf{x}$ is a *binary vector* of length n. For inner products or scalar products on a vector space, we will use the "$\cdot$" notation. That is, the inner product of $\boldsymbol{a}$ and $\boldsymbol{b}$ is $\boldsymbol{a} \cdot \boldsymbol{b}$. For the direct product of two vector spaces V_1 and V_2, we use "$\times$." That is, $V_1 \times V_2 = \{(\boldsymbol{a}, \boldsymbol{b}) \,|\boldsymbol{a} \in V_1, \boldsymbol{b} \in V_2\}$.

We will use $\oplus$, equivalent to the exclusive or (xor) operation, to represent addition in $\mathbb{F}_2$, and + will represent addition in characteristic 0. However, we will also use + to represent addition of elements in a finite field, even if in an extension field of characteristic 2, as is accustomed in mathematics. The composition of functions will be denoted by "$\circ$." Brackets "||" will represent both cardinality of sets and absolute value, including complex absolute value, though it will be made clear by the context which applies.

The ring of polynomials over $\mathbb{F}_p$ with n variables will be denoted by $\mathbb{F}_p[x_1, x_2, \ldots, x_n]$, with $\mathbb{F}_p[x_1, x_2, \ldots, x_n]/\langle p(x)\rangle$ representing the quotient ring modulo $p(x)$. We use $K \overset{n}{\hookrightarrow} L$ to represent a degree n field extension from a subfield K to an extension field L. The absolute trace function of a field extension of degree n to its base field will be denoted by Tr_n. That is, $\mathrm{Tr}_n : \mathbb{F}_{p^n} \to \mathbb{F}_p$, and $\mathrm{Tr}_n(x) = \sum_{i=0}^{n-1} x^{p^i}$. If the dimension is clear we may simply use Tr. For the qth primitive root of unity, we use ζ_q, where $\zeta_q = e^{\frac{2\pi i}{q}}$, for any q.

2.2 Boolean Functions

Boolean functions have been studied extensively in the context of cryptography. We provide here some necessary background before proceeding onto vectorial Boolean functions which form the critical nonlinear components of many encryption schemes. More detailed explanations and results on classical Boolean functions can be found in [3], [4], [5], [6], and [7].

Definition 2.2.1 *A Boolean function f in n variables is a map from $\mathbb{F}_2^n$ to $\mathbb{F}_2$,*

$$f : \mathbb{F}_2^n \to \mathbb{F}_2. \tag{2.1}$$

A Boolean function (BF) inputs a binary vector of length n and outputs a 0 or 1. There are multiple ways to represent a BF and depending on the context one may be more advantageous than another. An easy visual method to represent a BF is called the *truth table*, which is a (0,1)-sequence of the images of every element in $\mathbb{F}_2^n$. In other words, the truth table of f is $(f(x_0), f(x_1), \ldots, f(x_{2^n-1}))$, with $x_i \in \mathbb{F}_2^n$ ordered *lexicographically* in the natural counting order.

Example 2.2.1 *Table* 2.1 *is the truth table of a BF $f : \mathbb{F}_2^3 \to \mathbb{F}_2$. This low-dimension example function will serve to demonstrate cryptographic properties of Boolean functions as we proceed.*

x_3	x_2	x_1	f
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Table 2.1. Truth Table of a Boolean Function f

The *Hamming weight* of an arbitrary vector in $\mathbb{F}_2^n$, denoted by $wt(\boldsymbol{x})$, is the number of 1's in the vector $\boldsymbol{x}$. The Hamming weight of a BF is the Hamming weight of its truth table. For Example 2.2.1, we have $wt(f) = 4$. The *Hamming distance* between two arbitrary vectors in $\mathbb{F}_2^n$, denoted by $d(\boldsymbol{x}, \boldsymbol{y})$, is the weight of $\boldsymbol{x} \oplus \boldsymbol{y}$, i.e., $wt(\boldsymbol{x} \oplus \boldsymbol{y})$. The Hamming distance between two functions f and g is the Hamming weight of the difference in their truth tables, $wt(f \oplus g)$.

Another natural representation is a multivariate polynomial called the *algebraic normal form*. Because the inputs to the polynomial can only be 0 or 1, every variable must be considered modulo $x_i^2 - x_i$. Therefore, the algebraic normal form (ANF) is a polynomial in the quotient ring

$$\mathbb{F}_2[x_1, x_2, \ldots, x_n]/\langle x_1^2 - x_1, x_2^2 - x_2, \ldots, x_n^2 - x_n \rangle. \tag{2.2}$$

The ANF is given by

$$f(\boldsymbol{x}) = f(x_1, x_2, \ldots, x_n) = \sum_{\boldsymbol{a} \in \mathbb{F}_2^n} c_{\boldsymbol{a}} x_1^{a_1} x_2^{a_2} \cdots x_n^{a_n}, \tag{2.3}$$

where $c_{\boldsymbol{a}} \in \mathbb{F}_2$ and $\boldsymbol{a} = (a_1, a_2, \ldots, a_n) \in \mathbb{F}_2^n$.

The ANF of f as described in Table 2.1 is $f(\boldsymbol{x}) = f(x_1, x_2, x_3) = x_1x_3 \oplus x_1 \oplus x_2x_3$. The number of variables in the highest order monomial with nonzero coefficient is the *algebraic degree*. This definition of algebraic degree makes sense due to the existence and uniqueness of the ANF [5]. The algebraic degree of f is 2.

2.3 Cryptographic Properties of Boolean Functions

In order to be useful in a cryptographic algorithm, Boolean functions must have properties that realize the bedrock cryptographic principles of confusion and diffusion, described in the introductory chapter. Additionally, functions should have properties that defend against known attacks. Unfortunately, not all properties can be optimized at the same time, and therefore tradeoffs must be made. Because of the complexity in the relationships between some of the properties, there is no agreed upon standard of which properties at what level must be achieved. Cryptographic designers choose properties that meet their needs and provide resistance against the types of attacks believed most likely to be faced. One important property was already introduced, the algebraic degree. The algebraic degree must be high to avoid multiple known attacks, including higher-order differential cryptanalysis on block ciphers and attacks against low linear complexity on stream ciphers [7]. In this section, we describe several important properties of Boolean functions that prepare the reader for the follow-on chapters. For a more detailed analysis of cryptographic properties of Boolean functions, including tradeoffs, see [5], [7], [8].

2.3.1 Balance

A Boolean function is *balanced* if its output is half 0's and half 1's. Balance helps avoid statistical dependence between the input and the output and is widely considered an essential property. As a simple example of a tradeoff, the maximum algebraic degree of a n variable balanced function is $n - 1$. The function f from Table 2.1 is balanced.

2.3.2 Nonlinearity

Linear systems are often easy to solve, which is antithetical to the goal of a cryptographic transformation. A cryptographic Boolean function should therefore be highly nonlinear.

To measure nonlinearity, the distance of a BF from *any* linear or affine (i.e., degree ≤ 1) Boolean function must be measured. The *Walsh-Hadamard transform*, a version of the discrete Fourier transform, captures this information and much more.

Definition 2.3.1 *The Walsh-Hadamard transform (WHT) of a function f on $\mathbb{F}_2^n$ is the integer-valued function $W_f : \mathbb{F}_2^n \to \mathbb{Z}$ defined by*

$$W_f(\boldsymbol{w}) = \sum_{\boldsymbol{x} \in \mathbb{F}_2^n} (-1)^{f(\boldsymbol{x}) \oplus \boldsymbol{w} \cdot \boldsymbol{x}}. \tag{2.4}$$

The value of the transform at $\boldsymbol{w}$ is called the *Walsh-Hadamard coefficient* and the multi-set containing the list of all 2^n coefficients is called the *Walsh-Hadamard spectrum*. This spectrum contains essential information on properties of the Boolean function. For instance, if we consider the coefficient at $\mathbf{0}$, we have the sum $\sum_{\boldsymbol{x} \in \mathbb{F}_2^n} (-1)^{f(\boldsymbol{x})}$. If f is balanced, then the sum is half 1's and half -1's, which add up to 0. Thus, if the coefficient at $\mathbf{0}$ is 0, we know the function is balanced.

However, the most useful information provided by the WHT is the nonlinear properties of f. The max absolute value of the coefficients is called the *linearity* of f, as it represents the number of bits in the truth table that match an affine function's truth table. The *nonlinearity* is defined as the minimum distance between f and the class of all affine Boolean functions on n variables. This value is denoted by $\mathcal{N}_f$ and is calculated by

$$\mathcal{N}_f = 2^{n-1} - \frac{1}{2} \max_{\boldsymbol{u} \in \mathbb{F}_2^n} |W_f(\boldsymbol{u})|. \tag{2.5}$$

We now provide an example, using the same function from Table 2.1.

Example 2.3.1 *To find the WHT coefficients of our function f, we use Definition 2.3.1. At $\boldsymbol{w} = 000$ and $\boldsymbol{w} = 001$, we have*

$$W_f(000) = (-1)^0 + (-1)^1 + (-1)^0 + (-1)^1 + (-1)^0 + (-1)^0 + (-1)^1 + (-1)^1 = 0,$$
$$W_f(001) = (-1)^0 + (-1)^0 + (-1)^0 + (-1)^0 + (-1)^0 + (-1)^1 + (-1)^1 + (-1)^0 = 4.$$

The rest of the coefficients are $W_f(010) = 4, W_f(011) = 0, W_f(100) = 0, W_f(101) = 4, W_f(110) = -4, W_f(111) = 0$*, which gives us a WH spectrum of*

$$\{0, 4, 4, 0, 0, 4, -4, 0\}.$$

Thus, the nonlinearity of f is $\mathcal{N}_f = 2^2 - \frac{1}{2}|4| = 2$. In other words, if we change the truth table of f in two positions, we would arrive at an affine function. The WHT coefficients also tell us *which* affine function. Consider any of the input $\boldsymbol{w}$'s such that the absolute value of its WHT coefficient is equal to the max absolute value of the WHT spectrum. For instance, $\boldsymbol{w} = (001)$, with coefficient of 4. This vector corresponds to the linear function $g(x_1, x_2, x_3) = x_1$, with a truth table of $(0, 1, 0, 1, 0, 1, 0, 1)$. Notice if we change two spots in the truth table of f (Table 2.1), the truth tables would match. Thus, the WHT provides us with the nonlinearity and the closest affine approximation of the Boolean function.

While we used the WHT definition to calculate f's WH spectrum in this example, we note here that there are much more efficient ways to compute these values. An example is included in [9], in which the complexity is lowered from $O(2^{2n})$ in the naive definition-based method we used in Example 2.3.1, to $O(n2^n)$ in a fast Fourier transform-based algorithm ($O(\cdot)$ is the usual big-Oh notation).

Continuing with our discussion of nonlinearity, there are certain Boolean functions that achieve optimal nonlinearity. These functions are called *bent*, and in the Boolean function context we now define.

Definition 2.3.2 *An even variable Boolean function f is bent if*

$$\mathcal{N}_f = 2^{n-1} - 2^{n/2-1}. \tag{2.6}$$

In the case of bent functions, the absolute value of all WHT coefficients is $2^{n/2}$. For odd variable functions an exact upper bound on nonlinearity is still unknown for $n \geq 9$ [6].

Unfortunately, bent functions are never balanced and therefore are rarely used in cryptographic schemes, another example of tradeoffs between different criteria. However, the idea

of bentness, (i.e., optimal nonlinearity), is a fascinating topic with decades of research and many generalities and extensions into other fields of mathematics. We will return to the topic in the following chapters and generalize the idea of bentness using the c-differential.

2.3.3 Strict Avalanche and Propagation Criteria

The Boolean function property most closely associated with Shannon's diffusion principle is the propagation criterion. There are several equivalent ways to define this property, but we choose to use the discrete derivative, as derivative-based analysis will be used extensively in follow-on chapters. As we will show, the derivative provides a natural method to measure the behavior of a function as certain bits of the input are changed. The strict avalanche criterion is a particular case of the propagation criterion when only one bit is changed. The overall propagation behavior of a function is often called the *avalanche characteristics*. We start by defining the discrete derivative in the case of Boolean functions.

Definition 2.3.3 *Let f be a Boolean function. The derivative of f with respect to $\boldsymbol{a} \in \mathbb{F}_2^n$ is the function*

$$D_a f(\boldsymbol{x}) = f(\boldsymbol{x} \oplus \boldsymbol{a}) \oplus f(\boldsymbol{x}), \text{ for all } \boldsymbol{x} \in \mathbb{F}_2^n. \tag{2.7}$$

With the derivative defined, we can now define the strict avalanche criterion (SAC) and propagation criterion (PC). One of the equivalent definitions is based on probabilities. In essence, SAC means that if one input bit is changed, then all output bits should change with probability 1/2. The generalization of SAC is PC, in which multiple input bits are changed and each output bit still changes with probability 1/2.

Definition 2.3.4 *Let f be a Boolean function. Then f satisfies the* strict avalanche criterion *if, for all $\boldsymbol{a}$ such that $wt(\boldsymbol{a}) = 1$, the derivative $D_{\boldsymbol{a}} f(\boldsymbol{x})$ is balanced.*

Definition 2.3.5 *Let f be a Boolean function. Then f satisfies the* propagation criterion *of order k if, for all $\boldsymbol{a}$ such that $1 \leq wt(\boldsymbol{a}) \leq k$, the derivative $D_{\boldsymbol{a}} f(\boldsymbol{x})$ is balanced.*

In other words, f is PC(k) if for all $\boldsymbol{a}$ of weight $1, 2, \ldots, k$, the derivative of f with respect to $\boldsymbol{a}$ differs from f for half of the truth table. The connection to diffusion is obvious. Not

only does a small change in the input significantly change the output, but the output is changed in a measurably random way. Checking to see if all derivatives are balanced can be accomplished using another essential analytical tool, the crosscorrelation and the associated autocorrelation functions, which we now define.

Definition 2.3.6 *Let f and g be Boolean functions from $\mathbb{F}_2^n \to \mathbb{F}_2$ and $\boldsymbol{u} \in \mathbb{F}_2^n$. The* crosscorrelation *between f and g at $\boldsymbol{u}$ is*

$$\mathfrak{C}_{f,g}(\boldsymbol{u}) = \sum_{\boldsymbol{x} \in \mathbb{F}_2^n} (-1)^{f(\boldsymbol{x} \oplus \boldsymbol{u}) \oplus g(\boldsymbol{x})}, \tag{2.8}$$

and the corresponding autocorrelation *of f is*

$$\mathfrak{C}_f(\boldsymbol{u}) = \sum_{\boldsymbol{x} \in \mathbb{F}_2^n} (-1)^{f(\boldsymbol{x} \oplus \boldsymbol{u}) \oplus f(\boldsymbol{x})}. \tag{2.9}$$

Notice the exponent in the autocorrelation function is the derivative of f with respect to $\boldsymbol{u}$. Thus, if the derivative is balanced, half the values with be -1 and the other half 1, resulting in a value of 0 for the autocorrelation at $\boldsymbol{u}$. This gives us the following lemma.

Lemma 2.3.7 *A Boolean function f is PC(k) if and only if the autocorrelation is* 0 *for all $\boldsymbol{u}$ with $1 \leq wt(\boldsymbol{u}) \leq k$.*

We return to our example function f in Table 2.1 to demonstrate the autocorrelation function.

Example 2.3.2 *Using f's truth table of (0,1,0,1,0,0,1,1), we compute the autocorrelation as*

$$\mathfrak{C}_f(000) = (-1)^0 + (-1)^0 + (-1)^0 + (-1)^0 + (-1)^0 + (-1)^0 + (-1)^0 + (-1)^0 = 8,$$
$$\mathfrak{C}_f(001) = (-1)^1 + (-1)^1 + (-1)^1 + (-1)^1 + (-1)^0 + (-1)^0 + (-1)^0 + (-1)^0 = 0.$$

The rest of the autocorrelation is $\mathfrak{C}_f(010) = 0, \mathfrak{C}_f(011) = -8, \mathfrak{C}_f(100) = 0, \mathfrak{C}_f(101) = 0, \mathfrak{C}_f(110) = 0, \mathfrak{C}_f(111) = 0$, which gives us an autocorrelation spectrum

$$\{8, 0, 0, -8, 0, 0, 0, 0\}.$$

Notice the autocorrelation value at the vector $\mathbf{0}$ will always be 2^n; we call this the trivial autocorrelation. For f, all the inputs of weight 1 led to values of 0 in the autocorrelation, thus f is SAC, or equivalently, PC(1). However, the autocorrelation at (011) is 8 and therefore f is not PC(2).

If there exists a non-zero $\boldsymbol{u}$ such that $\mathfrak{C}_f(\boldsymbol{u})$ is $\pm 2^n$, then $\boldsymbol{u}$ is called a *linear structure* of f. Linear structures are considered weaknesses of cryptographic Boolean functions [10]. We will return to this topic in Chapter 3 from the perspective of the new c-derivatives.

2.3.4 Connection between Nonlinearity and Propagation Criterion

In [11] Meier and Staffelbach showed that Boolean functions that have all non-trivial autocorrelation values of 0 (i.e., PC(n) functions), are equivalent to the bent functions we introduced in Section 2.3.2. That is, optimal nonlinearity is equivalent to optimal propagation criterion and can only be achieved in unbalanced functions with an even number of variables.

Later, in [10], Nyberg called functions whose derivatives are balanced in all non-zero directions *perfect nonlinear*. Thus, if a Boolean function's all non-trivial autocorrelation coefficients are 0, the function is perfect nonlinear (PN), bent, and PC(n). Although we have seen these functions are not usable under most practical scenarios because bent functions are not balanced, having the derivatives *close to* balanced is a desirable feature. This notion will play a central role in Chapter 3 when we introduce the c-derivative and extend perfect nonlinearity in another direction.

2.3.5 Other Properties

We have introduced the properties of classical Boolean functions that will be expanded upon in the rest of this dissertation. However, there are many other properties that must be taken into account when considering Boolean functions to serve as primitives in cryptographic transformations. *Correlation immunity* measures the independence of the output of a Boolean function on any specific input variables and must be high in stream cipher applications. Functions that are correlation immune and balanced are called *resilient*. *Algebraic*

immunity, another important property of Boolean functions used in stream ciphers, measures resistance against algebraic attacks. The property called *Algebraic thickness* counts the minimum number of terms in any ANF of a function with which it is affinely equivalent, and should be high. Equivalence relations among Boolean functions will be discussed in the vectorial Boolean functions section.

There are many considerations when choosing a Boolean function as a cryptographic primitive. Some properties are generally considered to be essential regardless of the application: balancedness, high nonlinearity, and high degree. The other properties are more focused on the specific applications of the Boolean function.

2.4 Vectorial Boolean Functions

While Boolean functions have a rich history of research and application, many cryptographic transformations cannot be fully represented by one because of the single output bit. While a classical BF maps n bits into one bit, the nonlinear substitution boxes of block ciphers map n bits into m bits. These multi-bit output functions are called *vectorial Boolean functions*. We present the relevant background information here before moving onto the differential attack against block ciphers. A more detailed look into vectorial BFs can be found in [12].

Definition 2.4.1 *Let n and m be two positive integers. A* vectorial Boolean function F is a map from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$,

$$F : \mathbb{F}_2^n \to \mathbb{F}_2^m. \tag{2.10}$$

Boolean functions introduced in Section 2.2 coincide with $m = 1$ and are sometimes referred to as classical Boolean functions. Vectorial BFs from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ are also called (n, m)-functions, and we will use this notation throughout this dissertation. In a cryptographic context, they are often called substitution boxes, or "S-boxes," and are often the only nonlinear component of an encryption scheme. As opposed to the lower case f's we used for classical BFs, we will use upper case F's to represent vectorial BFs. Given an (n, m)-function F, with $\boldsymbol{x} \in \mathbb{F}_2^n$, and classical BFs $f_1, f_2, ..., f_m$, if $F(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ...f_m(\boldsymbol{x}))$, then $f_1, f_2, ...f_m$ are called the *coordinate functions* of F.

Similar to classical BFs, we first represent a vectorial BF with a truth table. In our example,

we will use a $(3, 3)$ function, but in general n does not always equal m. With 3-bit vector outputs for every input, it is typical to list the output as the result of three separate coordinate functions, as we have done in Table 2.2.

x_3	x_2	x_1	f_3	f_2	f_1
0	0	0	0	1	0
0	0	1	0	1	1
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	1	1	1
1	0	1	1	0	0
1	1	0	1	0	1
1	1	1	0	0	0

Table 2.2. Truth Table of a Vectorial Boolean Function F

While the coordinate functions of F must have good properties as described in Section 2.3, that is not sufficient. All non-zero *linear combinations* of these coordinate functions must also meet the desired properties. These linear combinations are called the *component functions* of F. If $\boldsymbol{b}$ is an element of $\mathbb{F}_2^m$, we can represent these component functions by the scalar product of $\boldsymbol{b}$ and F, i.e., the component functions of F are the classical Boolean functions $\boldsymbol{b} \cdot F(\boldsymbol{x})$, $\boldsymbol{b} \neq 0$.

Instead of viewing a vectorial BF in binary as a truth table of coordinate functions, we can also view the function in an integer (or hexadecimal) format, which is more representative of how an S-box is normally presented and can be seen in Table 2.3.

x	0	1	2	3	4	5	6	7
$F(x)$	2	3	1	6	7	4	5	0

Table 2.3. The S-box F

2.4.1 Finite Fields and Univariate Polynomial Representation

The notion of the algebraic normal form from classical BFs can be extended to vectorial BFs, but this representation will not be used in this dissertation. In the case where $m = n$, there is another representation, the *univariate polynomial representation*, that will be used extensively in the following chapters. In order to do this we first need to endow the vector space $\mathbb{F}_2^n$ with the structure of the finite field $\mathbb{F}_{2^n}$ [12]. This is possible because $\mathbb{F}_{2^n}$ can be viewed as a n-dimensional vector space over $\mathbb{F}_2$, with a natural basis composed of the powers of a primitive element. The elements of $\mathbb{F}_2^n$ are then the coefficients of the powers. Since our example function F from Table 2.2 is a $(3, 3)$ function, we provide an example of how to endow $\mathbb{F}_2^3$ with the structure of $\mathbb{F}_{2^3}$.

Example 2.4.1 . *Let an instance of $\mathbb{F}_{2^3}$ be defined by $\mathbb{F}_2[x]/\langle x^3 + x + 1\rangle$ with α being a root of $x^3 + x + 1$. Then Table 2.4 has the following mapping.*

$\mathbb{F}_{2^3}$	0	α^0	α^1	α^2	α^3	α^4	α^5	α^6
$\mathbb{F}_{2^3}$	0	1	α	α^2	$\alpha + 1$	$\alpha^2 + \alpha$	$\alpha^2 + \alpha + 1$	$\alpha^2 + 1$
$\mathbb{F}_2^3$	000	001	010	100	011	110	111	101
$\mathbb{Z}_8$	0	1	2	4	3	6	7	5

Table 2.4. A Mapping between $\mathbb{F}_{2^3}$ and $\mathbb{F}_2^3$. Adapted from [9].

With the ability to construct such a mapping, an (n, n)-function can be uniquely represented as a polynomial in one variable with coefficients in F_{2^n}, that is, the univariate polynomial

$$F(x) = \sum_{i=0}^{2^n-1} a_i x^i,\ a_i \in \mathbb{F}_{2^n}. \tag{2.11}$$

To find the coefficients a_i, we can use the discrete Fourier transform, as shown in [9]. Then we have $a_0 = F(0), a_{2^n-1} = \sum_{x\in\mathbb{F}_{2^n}} F(x)$, and

$$a_i = \sum_{k=0}^{2^n-2} F(\alpha^k)\alpha^{-ki}, 1 \le i \le 2^n - 2.$$

When we apply this technique to our example function F, we get the univariate polynomial $F(x) = x^3 + \alpha$. The algebraic degree of a function under the univariate polynomial representation is not the highest integer power, but rather the largest binary Hamming weight of the exponents i with $a_i \neq 0$. In our case, since $3 = 011$, $deg(F) = wt(011) = 2$.

With the univariate polynomial representation and the associated mapping between $\mathbb{F}_2^n$ and $\mathbb{F}_{2^n}$, it is sometimes more convenient to work in the finite field rather than the vector space. In other cases, the vector space may be better suited. It will be clear which environment we are working in based on the context.

2.4.2 Walsh Hadamard Transform and Autocorrelation for Vectorial Boolean Functions

The Walsh-Hadamard transform that was essential in analysis of the nonlinearity of classical BFs is now defined for (n, m)-functions.

Definition 2.4.2 *The Walsh-Hadamard transform of an (n, m)-function at $(\boldsymbol{a}, \boldsymbol{b})$, with $\boldsymbol{a} \in \mathbb{F}_2^n, \boldsymbol{b} \in \mathbb{F}_2^m$, is the Walsh-Hadamard transform of its component function $\boldsymbol{b} \cdot F(\boldsymbol{x})$ at $\boldsymbol{a}$. That is*

$$W_F(\boldsymbol{a}, \boldsymbol{b}) = \sum_{\boldsymbol{x} \in \mathbb{F}_2^n} (-1)^{\boldsymbol{b} \cdot F(\boldsymbol{x}) \oplus \boldsymbol{a} \cdot \boldsymbol{x}}. \tag{2.12}$$

The WH spectrum of a (n, m)-function can therefore be represented by an n by m array with the rows of $\boldsymbol{a} \in F_2^n$ and the columns of $\boldsymbol{b} \in F_2^m$. The WH spectrum array, also called a *linear approximation table*, for our example function F is in Table 2.5.

$\boldsymbol{a}\backslash\boldsymbol{b}$	000	001	010	011	100	101	110	111
000	8	0	0	0	0	0	0	0
001	0	-4	0	-4	0	4	0	-4
010	0	0	-4	4	0	0	-4	-4
011	0	4	-4	0	0	4	4	0
100	0	0	-4	-4	4	-4	0	0
101	0	4	4	0	4	0	0	-4
110	0	0	0	0	4	4	-4	4
111	0	4	0	-4	-4	0	-4	0

Table 2.5. WHT Table of a Vectorial Boolean Function F

This table contains all of the WH transforms of the component functions of F. We call the nonlinearity of F the *minimum* nonlinearity of any of its components' functions (recall we do not consider $\boldsymbol{b} = 0$ as a component function). With a max absolute value of 4 in the WHT table, from Equation 2.4 we know that the nonlinearity of F is therefore 2. Just as with classical BFs, the maximum possible nonlinearity is achieved for even number of variables when the nonlinearity is $2^{n-1} - 2^{n/2-1}$. In this case, every component function is bent, and we call the vectorial BF bent as well. There are restrictions on the values of n, m, in order for an (n, m)-function to be bent. In addition to n being even, it was shown in [10] that m cannot be larger than $n/2$. For this reason, there are no bent (n, n)-functions.

Next, the crosscorrelation from Definition 2.3.6 is extended to vectorial BFs.

Definition 2.4.3 . *Let F, G :* $\mathbb{F}_2^n \to \mathbb{F}_2^m$ *be* (n, m)*-functions. The* crosscorrelation *at* $\boldsymbol{u} \in \mathbb{F}_2^n, \boldsymbol{b} \in \mathbb{F}_2^m$ *is defined as*

$$\mathfrak{C}_{F,G}(\boldsymbol{u}, \boldsymbol{b}) = \sum_{\boldsymbol{x} \in \mathbb{F}_2^n} (-1)^{\boldsymbol{b} \cdot (F(\boldsymbol{x} \oplus \boldsymbol{u}) \oplus G(\boldsymbol{x}))} \tag{2.13}$$

and the corresponding autocorrelation *of F is*

$$\mathfrak{C}_F(\boldsymbol{u}, \boldsymbol{b}) = \sum_{\boldsymbol{x} \in \mathbb{F}_2^n} (-1)^{\boldsymbol{b} \cdot (F(\boldsymbol{x} \oplus \boldsymbol{u}) \oplus F(\boldsymbol{x}))}. \tag{2.14}$$

Just as we used an n by m array to display the WHT of an (n, m)-function, we also use an n by m array to display the autocorrelation table of our example function F from Table 2.2 in Table 2.6.

$\boldsymbol{u} \backslash \boldsymbol{b}$	000	001	010	011	100	101	110	111
000	8	8	8	8	8	8	8	8
001	8	-8	0	0	0	0	0	0
010	8	0	0	0	0	0	-8	0
011	8	0	0	0	0	0	0	-8
100	8	0	0	0	-8	0	0	0
101	8	0	0	0	0	-8	0	0
110	8	0	-8	0	0	0	0	0
111	8	0	0	-8	0	0	0	0

Table 2.6. Autocorrelation Table of a Vectorial Boolean Function F

The autocorrelation table displays the autocorrelation of the component functions of F, $\boldsymbol{b} \cdot F, \boldsymbol{b} \neq 0$. Recall that $\boldsymbol{u} = 0$ is the trivial autocorrelation.

Similar to how we defined the propagation criterion for a classical Boolean function, we use the equivalent notions of balanced derivatives and 0-valued autocorrelations (see Lemma 2.3.7) to define PC(k) for (n, m)-functions.

Definition 2.4.4 *Let F be an (n, m)-function. F satisfies the* propagation criterion *of order k if, for all $\boldsymbol{u}$ such that $1 \leq wt(\boldsymbol{u}) \leq k$, the derivative $D_u F(\boldsymbol{x})$ is balanced.*

Equivalently, if the autocorrelation function of F is 0 at all $\boldsymbol{u}$ with $1 \leq wt(\boldsymbol{u}) \leq k$, F is PC(k). A 0-valued autocorrelation, that is $\mathfrak{C}_F(\boldsymbol{u}, \boldsymbol{b}) = \sum_{\boldsymbol{x} \in \mathbb{F}_2^n} (-1)^{\boldsymbol{b} \cdot (F(\boldsymbol{x} \oplus \boldsymbol{u}) \oplus F(\boldsymbol{x}))} = 0$,

implies the $\boldsymbol{b}$-component functions of F have autocorrelation value of 0. Again, we see the (n, m)-function obtains a property only when all of the component functions do.

2.4.3 Differential Uniformity

The final property of vectorial Boolean functions we will introduce in this foundational chapter is directly related to differential cryptanalysis. The motivation for the property will be made clear in Section 2.5, where we introduce and describe differential cryptanalysis.

Definition 2.4.5 *Let F be an (n, m)-function with $\boldsymbol{a} \in \mathbb{F}_2^n, \boldsymbol{b} \in \mathbb{F}_2^m$, and let $\Delta_F(a, b) = |\boldsymbol{x} \in \mathbb{F}_2^n : F(\boldsymbol{x} \oplus \boldsymbol{a}) \oplus F(\boldsymbol{x}) = \boldsymbol{b}|$. The quantity $\delta_F = \max\{\Delta_F(a, b) : \boldsymbol{a} \in \mathbb{F}_2^n, \boldsymbol{b} \in \mathbb{F}_2^m, \boldsymbol{a} \neq 0\}$ is called the* differential uniformity *of F.*

To find the differential uniformity (DU) of an (n, m)-function, a *difference distribution table (DDT)* is constructed, with the rows consisting of the $\boldsymbol{a}$'s, columns consisting of the $\boldsymbol{b}$'s, and inputs consisting of $\Delta_F(a, b)$. We use our function from Table 2.2 to demonstrate in Table 2.7.

$\boldsymbol{a}\backslash\boldsymbol{b}$	000	001	010	011	100	101	110	111
000	8	0	0	0	0	0	0	0
001	0	2	0	2	0	2	0	2
010	0	0	2	2	2	2	0	0
011	0	2	2	0	2	0	0	2
100	0	0	0	0	2	2	2	2
101	0	2	0	2	2	0	2	0
110	0	0	2	2	0	0	2	2
111	0	2	2	0	0	2	2	0

Table 2.7. Difference Distribution Table of a Vectorial Boolean Function F

For $\boldsymbol{a} \neq 0$, we see the largest value is 2, and the differential uniformity of F, δ_F, is therefore 2.

The differential uniformity counts the number of $\boldsymbol{x}$'s in $\mathbb{F}_2^n$ such that, for specific $\boldsymbol{a}$ and $\boldsymbol{b}$ in F_2^n, the derivative of F with respect to $\boldsymbol{a}$ equals $\boldsymbol{b}$. Recall from Subsection 2.3.4 that balanced derivatives, or at least close to balanced derivatives, are a desirable property. We will discuss why in Section 2.5.

2.4.4 Equivalence Relations on Vectorial Boolean Functions

The number of Boolean and vectorial Boolean functions increases rapidly with the number of variables. There are 2^{2^n} classical BFs in n variables and $(2^m)^{2^n} = 2^{m2^n}$ vectorial BFs with n input variables and m output bits.

From Table 2.8, it is obvious searching through the entire space of BFs with many variables is infeasible, and the situation is even more dire for vectorial BFs. However, many of the properties we care most about remain invariant under certain transformations. These transformations form equivalence classes which partition the count of functions to more reasonable sizes. Functions with good properties, e.g., high nonlinearity, be can classified by these relations.

n	2	3	4	5	6	7	8	9
BFs' count	16	256	65,536	4,294,967,296	$\approx 1.8e19$	$\approx 3.4e38$	$\approx 1.1e77$	$\approx 1.3e154$

Table 2.8. Growth of Boolean Functions on n Variables

Definition 2.4.6 *Let F and G be (n, m)-functions. F and G are:*

1) Affine Equivalent *if $G = A \circ F \circ B$, where A and B are affine mappings of $\mathbb{F}_2^m$ and $\mathbb{F}_2^n$, respectively.*
2) Extended Affine Equivalent *if $G = A \circ F \circ B + C$, where A and B are affine mappings of $\mathbb{F}_2^m$, and $\mathbb{F}_2^n$, respectively, and C in an affine mapping from $\mathbb{F}_2^n \to \mathbb{F}_2^m$.*
2) Carlet-Charpin-Zinoviev Equivalent *if the graphs of G and F, $\{(\boldsymbol{x}, G(\boldsymbol{x}))|\boldsymbol{x} \in \mathbb{F}_2^n\}$ and $\{(\boldsymbol{x}, F(\boldsymbol{x}))|\boldsymbol{x} \in \mathbb{F}_2^n\}$, are equal under some affine mapping A of $\mathbb{F}_2^n \times \mathbb{F}_2^m$. That is, $(\boldsymbol{x}, G(\boldsymbol{x})) = A(\boldsymbol{x}, F(\boldsymbol{x}))$.*

Affine equivalence is a particular case of extended affine (EA) equivalence when $C = 0$. EA equivalence is in turn a particular case of Carlet-Charpin-Zinoviev (CCZ) equivalence. Differential uniformity is invariant under the CCZ equivalence [13], and is therefore invariant under EA and affine equivalences. Thus, if a function with good differential uniformity is, for instance, difficult to implement in hardware or software, an EA equivalent function could maintain the same DU but be easier to implement. Later, in Chapter 4, we will show that the new c-differential uniformity property does *not* hold in general under EA equivalence and a cipher designer may not be able to make a similar substitution.

2.5 Differential Cryptanalysis

In 1991, Biham and Shamir published details of a new cryptanalytic tool called differential cryptanalysis [14]. The target of the attack was the block cipher standard of the time, the Data Encryption Standard (DES). Although this was the first public appearance of the technique in print, one of the creators of DES at the company IBM would later state in [15] that the design team, along with the U.S. National Security Agency, were aware of the technique in the 1970s and DES was designed to be resistant against it. The same article, published in 1994, included rationale for why some of the design criteria of DES were not made public:

> disclosure of the design considerations would reveal the technique of differential cryptanalysis, a powerful technique that can be used against many ciphers. [15]

With a 56-bit key, a brute force attack on DES could require up to 2^{56} attempts before exhausting the key space. While this may have seemed like a large enough space at the time DES was introduced, the continual upward trend of computing power eventually rendered the DES algorithm vulnerable to brute force attacks, and a replacement was sought in the mid-1990s [16]. The differential cryptanalysis technique on DES reduces the complexity from the brute force attack, but still requires 2^{47} chosen plaintexts, an unrealistic amount in practical situations. While the differential attack on DES proved to be more theoretical than practical, the groundwork was laid for a very effective cryptanalytic tool that has evolved and now considered a primary concern of any block cipher.

2.5.1 Overview of an Attack

Modern block ciphers are iterative, that is, they are composed of multiple rounds. In order to realize confusion and diffusion, the rounds involve substitutions and permutations, with most achieving confusion through look up tables that are in reality the (n, m)-functions and S-boxes described in Section 2.4. While these S-boxes also have properties related to diffusion, other parts of block ciphers are primarily responsible to ensure diffusion is achieved. The two most popular ways to design a block cipher are the *Feistel network* and the *substitution permutation network (SPN)*, see [3] for a detailed description of both designs. Figure 2.1 shows an example 16-bit four round substitution permutation network block cipher diagram. In this basic example, the S's represent the S-boxes. The plaintext is xor'd with an initial key, broken into four parts, and then fed into four S-boxes. After the S-box, the outputs are mixed up via a permutation (diffusion) box and xor'd with a round key before the next round starts. Different encryption schemes use different numbers of rounds; most are at a minimum ten rounds and some have more than 30. The desired output is a ciphertext that appears random to an adversary without the secret key.

Assume we inject two plaintexts, p_1 and p_2, into a block cipher such that the difference between the two is Δp, i.e., $p_1 \oplus p_2 = \Delta p$. As the two plaintexts move through rounds of the cipher, they are transformed, resulting in ciphertexts c_1 and c_2. Let $\Delta c = c_1 \oplus c_2$ be the difference between the two ciphertexts. In an ideal cipher, the probability that a certain Δc occurs given a Δp would be $1/2^n$ where n is the number of bits in a block of plaintext. In other words, if $p_1 \oplus p_2 = \Delta p$ and $c_1 \oplus c_2 = \Delta c$, then any other pair of plaintexts that also have difference Δp have a different Δc. Anything less than this ideal is the vulnerability targeted in differential cryptanalysis.

The pair $(\Delta p, \Delta c)$ is called a *differential*. The classical differential attack is a statistical chosen-plaintext attack in which the attacker chooses plaintexts with difference Δp and seeks to gain insight into the secret key by exploiting the fact that Δc occurs with high probability, compared to $1/2^n$. Notice the difference Δp is *not* changed by the xor operation with the round key because $(p_1 \oplus k) \oplus (p_2 \oplus k) = p_1 \oplus p_2 \oplus k \oplus k = \Delta p$. However, Δp *is* changed by the S-box. This is where our look into the difference distribution table (Table 2.7 is an example) and differential uniformity becomes relevant. Recalling the equation used to find the differential uniformity, $F(\boldsymbol{x} \oplus \boldsymbol{a}) \oplus F(\boldsymbol{x}) = \boldsymbol{b}$, we see the difference in input is $\boldsymbol{a}$

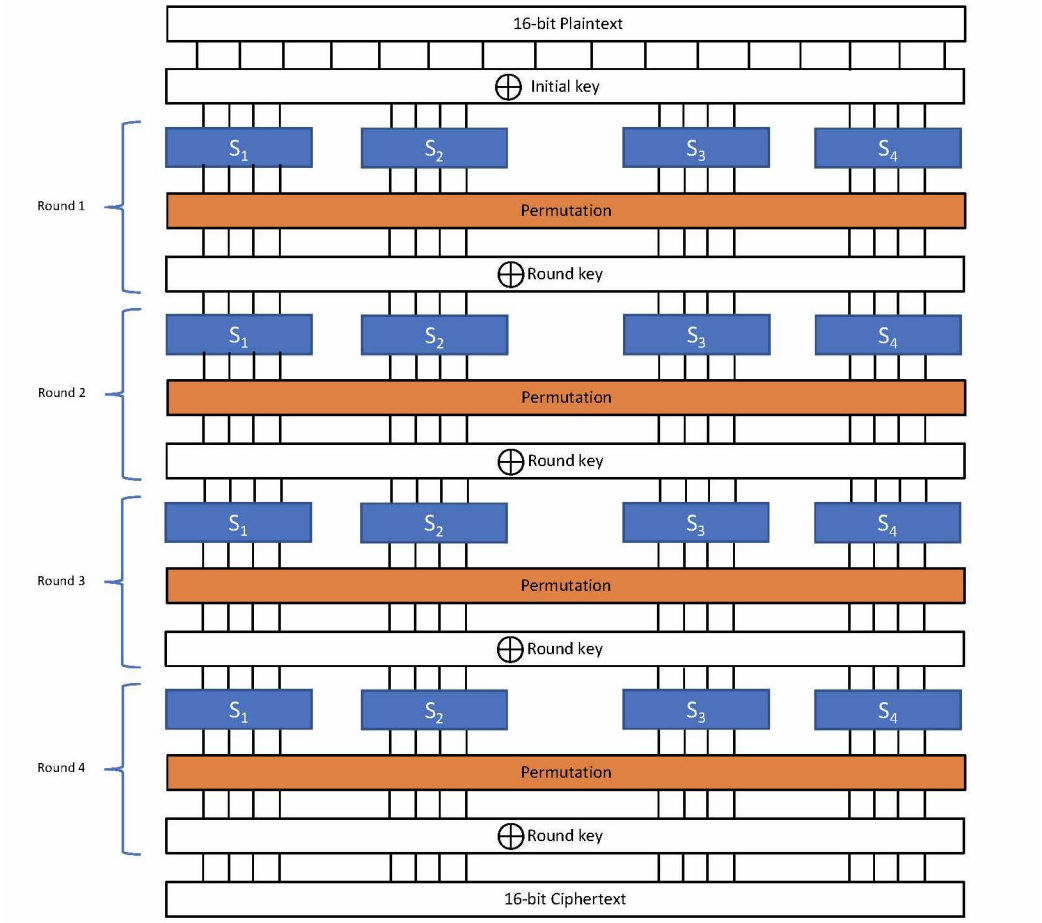

Figure 2.1. 4-Round Substitution Permutation Network (SPN)

while the difference in output is $\boldsymbol{b}$. That is, for one S-box, $(\boldsymbol{a}, \boldsymbol{b}) = (\Delta p, \Delta c)$.

Of course, there is more than one S-box and there are many rounds. The output of the high probability differential from one S-box to the next is used, and a path is traced through the cipher, using the highest probabilities available from DDT's of the different S-boxes. The probabilities from all the rounds, except for the last round, are combined to determine the *differential characteristic*. If this differential characteristic is significantly larger than the brute force probability (i.e., exhaustive key search), then a differential attack may reduce the complexity of the key search or "break" the cipher.

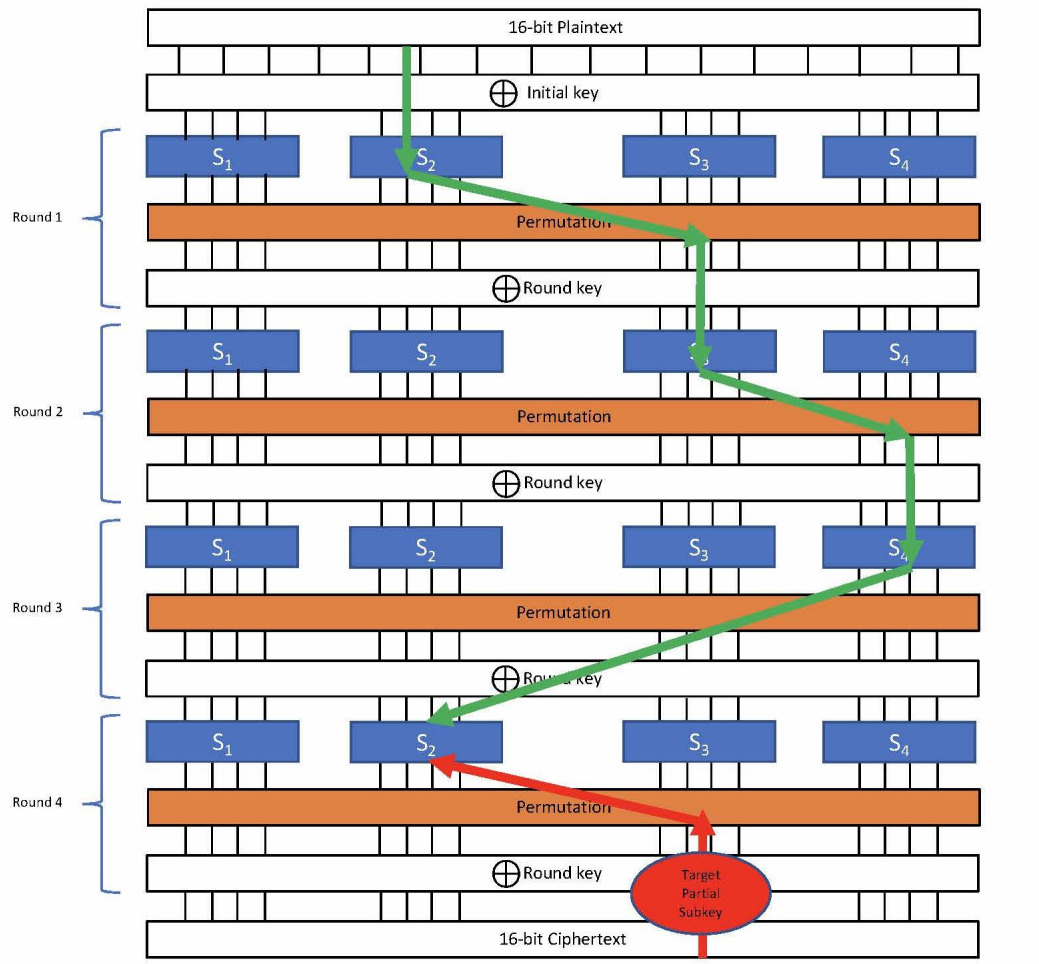

Figure 2.2. SPN with Differential Trace

The cryptanalysis process will involve the encryption of many chosen plaintexts that satisfy the Δp difference identified while determining the differential characteristic. The attack then moves onto attempted recovery of bits from the last round key. The specific design of each block cipher requires modifications to this description, often times making the attack much more sophisticated, see [17] for a more detailed description of an attack. But in general, using ciphertexts from known plaintexts, a partial decryption is performed on the bits corresponding to the target last round S-boxes, with the corresponding key bits called the *target partial subkey*. The target partial subkey is small enough that all combinations can be attempted for the ciphertexts. After this partial decryption, the bits are run backward

through the S-box of the final round to see if they match the high probability expected difference. If they do, they are deemed *good pairs*. The target partial subkey attempt that gives us the highest count of good pairs is probably correct. If so, the attack has extracted key bits from the last round key and can be used to further attack the system. Figure 2.2 provides a simplified visual simulation of the process.

2.5.2 Resistance against a Differential Attack

Based on the description of the attack, it is obvious that the primary mitigation for an iterated block cipher against a differential attack is ensuring that there are no differential characteristics that result in probabilities much better than exhaustive key search. This can be achieved by ensuring low differential uniformity for S-boxes and having a high number of rounds with many permutations. In fact, in [18] the authors demonstrate that under certain conditions this resistance can be proven.

Keeping differential uniformity low is now considered essential in all block cipher designs. Because differential uniformity is defined using derivatives, the question of optimally low differential uniformity is tied to Nyberg's notion of perfect nonlinearity and balanced derivatives. In Subsection 2.4.2 it was observed that there are no bent (n, n)-functions and in Subsection 2.3.4 it was observed that bentness and balanced derivatives are equivalent for classical Boolean functions. From Section 2.4 we know all component functions of (n, n)-functions are classical Boolean functions. Thus, there are no perfect nonlinear (n, n)-functions.

Another way to demonstrate an (n, n)-function cannot have balanced derivatives (and therefore not be perfect nonlinear) is by using the property that if $\boldsymbol{x}$ is a solution to $F(\boldsymbol{x} \oplus \boldsymbol{a}) \oplus F(\boldsymbol{x}) = \boldsymbol{b}$, then so is $\boldsymbol{x} \oplus \boldsymbol{a}$. But in an (n, n)-function, in order to be balanced the derivative must achieve every value of $\boldsymbol{b}$ only once. The best achievable DU is therefore when $\boldsymbol{x}$ and $\boldsymbol{x} \oplus \boldsymbol{a}$, for some $\boldsymbol{a}$, are the only solutions for any $\boldsymbol{b}$. If this is the case the (n, n)-function has DU of 2 and is called *almost perfect nonlinear (APN)*. The "almost" is a bit unfortunate because it suggests something less than optimal. However, in the case of an (n, n)-function, APN is optimal and results in the lowest DU possible.

2.5.3 Extensions of Differential Cryptanalysis

The original differential cryptanalysis technique has been modified and extended in multiple ways and differential cryptanalysis is now considered a class of attacks that exploit how any kind of difference can be traced through an encryption scheme to collect statistically relevant information on key bits. While some of these extensions apply to hashing algorithms and other types of symmetric encryption, our focus is on block ciphers. In [19], two extensions are introduced; *truncated* and *higher order* differential attacks. A truncated differential differs from the original notion in that only a subset of the bits of plaintext and ciphertext pairs are predicted as opposed to the entire bit strings. This can allow for higher probability differential characteristics and therefore higher likelihood of key bit recovery. Higher order differentials utilize the notion of higher order discrete derivatives in vectorial Boolean functions, introduced and explored in [20]. As opposed to the ordinary differentials that trace one plaintext at a time, these higher order differentials trace the differences of multiple plaintexts and have been shown to break ciphers that are traditionally resistant to ordinary differential cryptanalysis [21]. This idea will be revisited extensively in Chapter 5. In *impossible* differential cryptanalysis, instead of differential characteristics that have high probabilities, characteristics with *zero* probability are used to rule out many keys [22]. The *boomerang attack*, introduced in [23], allows an attacker to trace differentials only partly through a cipher, increasingly the likelihood of success on some ciphers.

These modifications and others have demonstrated that resistance against conventional differential cryptanalysis is necessary, but not always sufficient. Further steps must be taken to ensure resistance against all known extensions of differential cryptanalysis. It is almost certain that cryptanalysts are tweaking and enhancing these modifications while also attempting the creation of new techniques. In the next chapter, we begin our theoretical and practical investigation of a newly proposed differential and attempt to determine its relevance to the S-boxes of modern block ciphers.

2.6 Binary to P-ary

While the definitions, properties, and examples in this chapter up to this point are all in fields of characteristic 2 (i.e., binary), most can be generalized to fields of characteristic p, for any prime number p. In this section, we provide the more general characteristic p definitions of

the most relevant concepts that will be expanded upon in the following chapters. We start with the characteristic p generalization of Boolean functions, called p-ary functions.

Definition 2.6.1 *Let p be a prime number. A p-ary function f in n variables is a map from $\mathbb{F}_p^n$ to $\mathbb{F}_p$*

$$f : \mathbb{F}_p^n \to \mathbb{F}_p. \tag{2.15}$$

Recall in the binary case a Boolean function is balanced if the truth table has half 0's and half 1's. In n variables, this means 2^{n-1} values will be 1, and 2^{n-1} values will be 0. Extending to the p-ary case, we see a balanced p-ary function achieves each of the p different potential outputs p^{n-1} times. We can also generalize the notion of Hamming weight from binary to p-ary by considering how many non-zero positions are in a vector. That is, if $\boldsymbol{x}$ is a p-ary vector, then $wt(\boldsymbol{x})$ is the number of non-zero positions of $\boldsymbol{x}$.

The Walsh-Hadamard transform that was used to calculate the nonlinearity (and other properties) of a Boolean function in Subsection 2.3.2 is extended to the p-ary case by changing the base of the exponent from -1 to the more general pth primitive root of unity, ζ_p. Notice the $\oplus$ operation used in binary is replaced by the appropriate + or - modulo the prime characteristic.

Definition 2.6.2 *The Walsh-Hadamard transform of a p-ary function f on $\mathbb{F}_p^n$ is the complex-valued function $W_f : \mathbb{F}_p^n \to \mathbb{C}$ defined by*

$$W_f(\boldsymbol{w}) = \sum_{\boldsymbol{x} \in \mathbb{F}_p^n} \zeta_p^{f(\boldsymbol{x}) - \boldsymbol{w} \cdot \boldsymbol{x}}. \tag{2.16}$$

Next, we define the characteristic p generalization of (n, m) or vectorial Boolean functions.

Definition 2.6.3 *Let n and m be two positive integers and p be a prime number. A* vectorial *p*-ary function F is a map from $\mathbb{F}_p^n$ to $\mathbb{F}_p^m$,

$$F : \mathbb{F}_p^n \to \mathbb{F}_p^m. \tag{2.17}$$

We will call vectorial p-ary functions (n, m, p)-functions when p can be any prime, and (n, m)-functions when $p = 2$. Next, the derivative of a Boolean function is generalized to the p-ary case.

Definition 2.6.4 *Let f be a p-ary function. The derivative of f with respect to $\boldsymbol{a} \in \mathbb{F}_p^n$ is the function*

$$D_a f(\boldsymbol{x}) = f(\boldsymbol{x} + \boldsymbol{a}) - f(\boldsymbol{x}), \text{ for all } \boldsymbol{x} \in \mathbb{F}_p^n. \tag{2.18}$$

The crosscorrelation and autocorrelation functions are generalized to p-ary with this derivative, and as in the case of the p-ary WHT, a change of base of the exponent to ζ_p.

Definition 2.6.5 *Let $F, G : \mathbb{F}_p^n \to \mathbb{F}_p^m$ be (n, m, p)-functions. The* crosscorrelation *at $\boldsymbol{u} \in \mathbb{F}_p^n, \boldsymbol{b} \in \mathbb{F}_p^m$ is defined as*

$$\mathfrak{C}_{F,G}(\boldsymbol{u}, \boldsymbol{b}) = \sum_{\boldsymbol{x} \in \mathbb{F}_p^n} \zeta_p^{\boldsymbol{b} \cdot (F(\boldsymbol{x}+\boldsymbol{u}) - G(\boldsymbol{x}))} \tag{2.19}$$

and the corresponding autocorrelation *of F is*

$$\mathfrak{C}_F(\boldsymbol{u}, \boldsymbol{b}) = \sum_{\boldsymbol{x} \in \mathbb{F}_p^n} \zeta_p^{\boldsymbol{b} \cdot (F(\boldsymbol{x}+\boldsymbol{u}) - F(\boldsymbol{x}))}. \tag{2.20}$$

The Walsh-Hadamard transform of an (n, m, p)-function follows by calculating the WHT of its component p-ary functions.

Definition 2.6.6 *The Walsh-Hadamard transform of an (n, m, p)-function at $(\boldsymbol{a}, \boldsymbol{b})$, with $\boldsymbol{a} \in \mathbb{F}_p^n, \boldsymbol{b} \in \mathbb{F}_p^m$, is the Walsh-Hadamard transform of its component function $\boldsymbol{b} \cdot F(\boldsymbol{x})$ at $\boldsymbol{a}$. That is,*

$$W_F(\boldsymbol{a}, \boldsymbol{b}) = \sum_{\boldsymbol{x} \in \mathbb{F}_p^n} \zeta_p^{\boldsymbol{b} \cdot F(\boldsymbol{x}) - \boldsymbol{a} \cdot \boldsymbol{x}}. \tag{2.21}$$

The vector space to finite field mapping described in Subsection 2.4.1, which is essential for the rest of this dissertation, also works for $\mathbb{F}_p^n$ to $\mathbb{F}_{p^n}$. We provide an example in the case of $\mathbb{F}_3^2$.

Example 2.6.1 *Let an instance of* $\mathbb{F}_{3^2}$ *be defined by* $\mathbb{F}_3[x]/\langle x^2+2x+2\rangle$ *with* α *being a root of* x^2+2x+2*. Then Table 2.9 has the following mapping.*

$\mathbb{F}_{3^2}$	0	α^0	α^1	α^2	α^3	α^4	α^5	α^6	α^7
$\mathbb{F}_{3^2}$	0	1	α	$\alpha+1$	$2\alpha+1$	2	2α	$2\alpha+2$	$\alpha+2$
$\mathbb{F}_3^2$	00	01	10	11	21	02	20	22	12
$\mathbb{Z}_9$	0	1	3	4	7	2	6	8	5

Table 2.9. A Mapping between $\mathbb{F}_{3^2}$ and $\mathbb{F}_3^2$

As in the binary case, when $m = n$, a function can be represented by a univariate polynomial of the form

$$F(x) = \sum_{i=0}^{p^n-1} a_i x^i,\ a_i \in \mathbb{F}_{p^n}. \tag{2.22}$$

Using the p-ary derivative, differential uniformity is extended into characteristic p.

Definition 2.6.7 *Let* F *be an* (n,m,p)*-function with* $\boldsymbol{a} \in \mathbb{F}_p^n, \boldsymbol{b} \in \mathbb{F}_p^m$*, and let* $\Delta_F(a,b) = |\boldsymbol{x} \in \mathbb{F}_p^n : F(\boldsymbol{x}+\boldsymbol{a}) - F(\boldsymbol{x}) = \boldsymbol{b}|$*. The quantity* $\delta_F = \max\{\Delta_F(a,b) : \boldsymbol{a} \in \mathbb{F}_p^n, \boldsymbol{b} \in \mathbb{F}_p^m, \boldsymbol{a} \neq 0\}$ *is called the* differential uniformity *of* F*.*

Unlike the case of $p = 2$, when $p > 2$ there do exist perfect nonlinear functions when $n = m$. That is, there exist (n,n,p)-functions with balanced derivates and differential uniformity of 1.

Example 2.6.2 . *The function* $F : \mathbb{F}_{3^2} \to \mathbb{F}_{3^2}$*, where* $F(x) = x^2$ *in univariate polynomial representation is perfect nonlinear. To see this, consider the derivative of* F*,* $D_aF(x) = (x+a)^2 - x^2 = x^2 + 2ax + a^2 - x^2 = 2ax + a^2$*. For all* $a \neq 0$*, the derivative is a permutation of the elements of* $\mathbb{F}_{3^2}$*, and therefore balanced. Because* F *has balanced derivatives for all* $a \neq 0$*,* F *is a perfect nonlinear* $(2,2,3)$ *function.*

The real-world S-boxes that will be investigated later in this dissertation are all in characteristic 2, as binary implementations continue to dominate the universe of block ciphers that have been made public. However, most of the mathematical results in Chapters 3, 4, and 5 will apply to any prime characteristic, and this final section gives us the tools to proceed in these environments.

To conclude this foundational chapter, we note that for reasons of speed and efficiency, the only nonlinear component of most symmetric block ciphers are the substitution boxes composed of (n, m)-functions that we have described. Thus, the S-box plays a uniquely important role in cryptography. With the essential definitions, notations, properties, and background in place, we proceed in our investigation into c-differentials and their relevance to these critical components of modern digital communications security.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:
c-Differentials and Generalizations of Cryptographic Properties of Vectorial Boolean Functions

As described in Chapter 2, differential cryptanalysis poses a real and evolving threat to block ciphers that utilize S-boxes. As one of the main potential vulnerabilities of a secret key encryption scheme, newly proposed ciphers often provide an analysis of resistance to the various known differential attacks, and these attacks continue to advance in sophistication. In [24], the authors defined and used a multiplicative differential against some real-world ciphers in an attempt to attack schemes that use modular multiplication as a primitive operation. Given an (n, m, p)-function $F : \mathbb{F}_{p^n} \to \mathbb{F}_{p^m}$ with p prime, instead of considering inputs $(x, x + a)$ with associated outputs $(F(x), F(x + a))$, the differential they introduced has inputs of the form (cx, x) and outputs $(F(cx), F(x))$. This technique proved successful against several variants of the IDEA cipher, and the authors challenged others to find new differentials.

Motivated by their success, Stănică et al. started a theoretical analysis of a different multiplicative differential in [2]. This new differential uses the same inputs as the original attack, $(x, x + a)$, but multiplies one of the outputs by c, with c residing in the codomain of F. This results in the outputs $(cF(x), F(x + a))$ and is called a "c-differential." Notice when $c = 1$ the original differential is recovered. Because the c-differential is multiplicative, it is more convenient to work with (n, m, p)-functions over finite fields instead of vector spaces. The normal vector space inner product used in Chapter 2 will be replaced by the absolute trace function of a field extension, $\mathrm{Tr}_n(x)$. That is, $a \cdot b \to \mathrm{Tr}_n(ab)$. With this new differential, the authors of [2] defined a new discrete derivative.

Definition 3.0.1 *Let $F : \mathbb{F}_{p^n} \to \mathbb{F}_{p^m}$ be an (n, m, p)-function with $c \in \mathbb{F}_{p^m}$. The* (multiplicative) c-derivative *of F with respect to $a \in \mathbb{F}_{p^n}$ is the function*

$$ {}_cD_aF(x) = F(x + a) - cF(x), \text{ for all } x \in \mathbb{F}_{p^n}. $$

Many essential cryptographic properties of vectorial Boolean and p-ary functions can be described in terms of the discrete derivative. In this chapter, we use the c-derivative to extend the notions of perfect nonlinearity, bentness, and avalanche characteristics by introducing a new crosscorrelation function and corresponding autocorrelation function. The material in sections 3.1, 3.2, and 3.3 is based on and expanded from Stănică, Gangopadhyay, Geary, Riera, and Tkachenko [25].

3.1 c-Crosscorrelation and c-Autocorrelation

In Section 2.4.2, the crosscorrelation and autocorrelation functions were defined for vectorial Boolean functions. Using the discrete c-derivative from Definition 3.0.1 we now extend these notions with the following new definitions.

Definition 3.1.1 *Let $F, G : \mathbb{F}_{p^n} \to \mathbb{F}_{p^m}$ be (n, m, p)-functions with $c \in \mathbb{F}_{p^m}$. The* c-crosscorrelation *at $u \in \mathbb{F}_{p^n}, b \in \mathbb{F}_{p^m}$ is defined as*

$$ {}_c\mathfrak{C}_{F,G}(u, b) = \sum_{x \in \mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(x+u)-cG(x)))} \tag{3.1} $$

and the corresponding c-autocorrelation *of F is*

$$ {}_c\mathfrak{C}_{F}(u, b) = \sum_{x \in \mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(x+u)-cF(x)))}. \tag{3.2} $$

Notice that when $c = 1$, $p = 2$, and we use the normal vector space inner product, the original (n, m)-function cross and autocorrelations are recovered. Working in finite fields, the component functions of F are $\mathrm{Tr}_m(bF), 0 \neq b \in \mathbb{F}_{p^m}$. Thus, the c-crosscorrelation of the (n, m, p)-function is the crosscorrelation of the b-component Boolean function of F with the c multiple of the b-component Boolean function G. That is, ${}_c\mathfrak{C}_{F,G}(u, b) = \mathfrak{C}_{\mathrm{Tr}_m(bF),\mathrm{Tr}_m(bcG)}(u)$. The same observation holds for the c-autocorrelation, ${}_c\mathfrak{C}_{F}(u, b) = \mathfrak{C}_{\mathrm{Tr}_m(bF),\mathrm{Tr}_m(bcF)}(u)$.

Throughout this chapter, we will be using the fact that balanced c-derivatives in the direction of u imply a c-autocorrelation of 0 at u. That is, if $F(x + u) - cF(x)$ is balanced, then

$\sum_{x\in\mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(x+u)-cF(x)))} = 0$. This follows from [26, Theorem 5.4], from which we see the exponential sum of a balanced function is 0.

Using our new autocorrelation function, we now proceed to generalize the definitions of perfect nonlinearity and bentness, show these two generalizations are equivalent notions, and capture the current definitions as special cases of our generalizations.

3.2 Perfect c-Nonlinearity

In [10], Nyberg defined a perfect nonlinear (n, m)-function as a function whose derivatives are balanced in every non-zero direction. In other words, the derivatives take every value the same number of times. For an (n, m, p)-function, this equates to the derivatives achieving every output p^{n-m} times, which only makes sense when $n \geq m$. As pointed out in the previous section, having balanced derivatives implies that the function's autocorrelation must be zero in all non-trivial cases. Recall from Section 2.4.2 that trivial autocorrelations occur when b or u are 0, in which case the autocorrelation of an (n, m, p)-function is p^n. With our new c-autocorrelation and $c \neq 1$, the trivial autocorrelations are when $b = 0$, which also results in p^n, but $u = 0$ is no longer a trivial case. We now extend Nyberg's definition into our notion of perfect c-nonlinearity.

Definition 3.2.1 *Let F be an (n, m, p)-function and $c \in \mathbb{F}_{p^m}$ fixed, we say that F is* perfect c-nonlinear (PcN) *if its c-autocorrelation ${}_c\mathfrak{C}_F(u, b) = 0$, for all $u \in \mathbb{F}_{p^n}^*$, $b \in \mathbb{F}_{p^m}^*$. A* strictly perfect c-nonlinear *is a function F for which all ${}_c\mathfrak{C}_F(u, b) = 0$, for all $u \in \mathbb{F}_{p^n}$, $b \in \mathbb{F}_{p^m}^*$.*

We show in Lemma 3.2.2 that, when $n = m$, PcN and strictly PcN are equivalent. Additionally, for $p = 2$ it was demonstrated in [27] that the c-derivative of a PcN function must be a permutation polynomial (and therefore balanced) when $c \neq 1$. Thus, PcN and strictly PcN are equivalent notions in even characteristic or when $n = m$, and the only case in which an (n, m, p)-function may be PcN but not strictly PcN is when $p > 2$ and $n \neq m$.

If the c-derivatives of an (n, m, p)-function are balanced, that is, if ${}_cD_aF(x) = F(x + a) - cF(x)$ at every fixed $a \neq 0$ assumes the same value $y \in \mathbb{F}_{p^m}$ for exactly p^{n-m} values of $x \in \mathbb{F}_{p^n}$, then F's nontrivial autocorrelation is 0 and F is perfect c-nonlinear. The converse is true if $n = m$, as shown in the following lemma.

Lemma 3.2.2 *Let F be an (n, n, p)-function. If F is perfect c-nonlinear, then the c-derivatives of F are balanced.*

Proof: Because F is PcN, the c-autocorrelation of F, ${}_c\mathfrak{C}_F(u, b)$, is 0 for $b \neq 0$. That is,

$$ {}_c\mathfrak{C}_F(u, b) = \sum_{x \in \mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(x+u)-cF(x)))} = 0. $$

Here we will use [26, Theorem 7.7], which tells us that if

$$ \sum_{x \in \mathbb{F}_{p^n}} \chi(f(x))) = 0, $$

then $f(x)$ is a permutation polynomial where χ is the canonical additive character of $\mathbb{F}_{p^n}$, which is $\chi_b(a) = \zeta_p^{\mathrm{Tr}_n(ba)}$. Then, we have

$$ \begin{aligned} 0 &= \sum {}_c\mathfrak{C}_F(u, b) \\ &= \sum_{x \in \mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_n(b(F(x+u)-cF(x)))} \\ &= \sum_{x \in \mathbb{F}_{p^n}} \chi_b((F(x+u) - cF(x)) \\ &= \sum_{x \in \mathbb{F}_{p^n}} \chi_b({}_cD_uF(x)). \end{aligned} $$

Then, by [26, Theorem 7.7], ${}_cD_uF(x)$ is a permutation polynomial, and the derivatives must be balanced. ■

Because the c-derivative of a PcN (n, n, p)-function is a permutation polynomial, when $n = m$ a PcN function is strictly PcN. Later, we show that a general (n, m, p)-function is perfect c-nonlinear if and only if the traces of the c-differentials are balanced.

3.3 c-Differential Bent

The definitions of bent Boolean and vectorial Boolean functions in Chapter 2 have been generalized in numerous ways. We mention here [5] and [6], which include many examples.

One generalization is from binary to p-ary for prime p. A p-ary function $f : \mathbb{F}_{p^n} \to \mathbb{F}_p$ is bent if the square of the complex absolute value of the Walsh Hadamard transforms is constant at all inputs, namely, $|W_f(w)|^2 = W_f(w)\overline{W_f(w)} = \mathfrak{C}_F(0) = p^n$, for all $w \in \mathbb{F}_{p^n}$. Notice this definition captures the classical bent Boolean functions, which have WHT spectrum of $\pm 2^{n/2}$, as $|2^{n/2}|^2 = 2^n = p^n$, since $p = 2$. This notion of bentness is extended to vectorial p-ary functions by simply considering the same product for each component function.

Using this idea, we now define a new bent concept that takes into account the c-differential.

Definition 3.3.1 *Let F be an (n, m, p)-function. F is* c-differential bent *if $W_F(a, b)\overline{W_F(a, bc)} = {}_c\mathfrak{C}_F(0, b)$, for all $a \in \mathbb{F}_{p^n}, b \in \mathbb{F}^*_{p^m}$.*

In other words, for each component function of F, if the product $W_F(a, b)\overline{W_F(a, bc)}$ is equal to the c-autocorrelation at zero for all inputs a, then F is c-differential bent. For $c = 1$, this is the classical case and well known. Our task is to show that the perfect c-nonlinearity and c-differential bent are equivalent. To show for all c we first need the following lemma, which is a modification of a known result [28] utilizing our new c-crosscorrelation and c-differential bent notion.

Lemma 3.3.2 *Let p be a prime number, F and G be (n, m, p)-functions, and $c \in \mathbb{F}_{p^m}$. Then for all $b \in F_{p^m}$, we have*

$$\begin{aligned} \sum_{u \in \mathbb{F}_{p^n}} {}_c\mathfrak{C}_{F,G}(u, b)\zeta_p^{-\mathrm{Tr}_n(ua)} &= W_F(a, b)\overline{W_G(a, bc)}, \\ {}_c\mathfrak{C}_{F,G}(u, b) &= p^{-n} \sum_{a \in \mathbb{F}_{p^n}} W_F(a, b)\overline{W_G(a, bc)}\zeta^{\mathrm{Tr}_n(ua)}. \end{aligned} \tag{3.3}$$

In particular, if $F = G$, then

$$\begin{aligned} \sum_{u \in \mathbb{F}_{p^n}} {}_c\mathfrak{C}_F(u, b)\zeta_p^{-\mathrm{Tr}_n(ua)} &= W_F(a, b)\overline{W_F(a, bc)} \\ {}_c\mathfrak{C}_F(u, b) &= p^{-n} \sum_{a \in \mathbb{F}_{p^n}} W_F(a, b)\overline{W_F(a, bc)}\zeta_p^{\mathrm{Tr}_n(ua)}. \end{aligned}$$

Proof: We start with

$$\begin{aligned}
&\sum_{u\in\mathbb{F}_{p^n}} {}_c\mathfrak{C}_{F,G}(u,b)\zeta_p^{-\mathrm{Tr}(ua)} = \sum_{u\in\mathbb{F}_{p^n}}\sum_{z\in\mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(z+u)-cG(z))}\zeta_p^{\mathrm{Tr}_n(-ua)}\\
=&\sum_{u\in\mathbb{F}_{p^n}}\sum_{z\in\mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(z+u)-cG(z))}\zeta_p^{-\mathrm{Tr}((z+u)a)+\mathrm{Tr}_n(za)}\\
=&\sum_{z\in\mathbb{F}_{p^n}} \zeta_p^{-\mathrm{Tr}_m(bcG(z))}\zeta_p^{\mathrm{Tr}_n(za)}\sum_{u\in\mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(bF(z+u))}\zeta_p^{-\mathrm{Tr}_n((z+u)a)}\\
\overset{w:=z+u}{=}&\sum_{z\in\mathbb{F}_{p^n}} \zeta_p^{-\mathrm{Tr}_m(bcG(z))}\zeta_p^{\mathrm{Tr}(za)}\sum_{w\in\mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(bF(w))}\zeta_p^{-\mathrm{Tr}_n(wa)}\\
=&W_F(a,b)\overline{W_G(a,bc)}.
\end{aligned}$$

For the second identity, we reverse the argument, and obtain

$$\begin{aligned}
&p^{-n}\sum_{a\in\mathbb{F}_{p^n}} W_F(a,b)\overline{W_G(a,bc)}\zeta^{\mathrm{Tr}_n(ua)}\\
=&p^{-n}\sum_{a\in\mathbb{F}_{p^n}}\sum_{z,w\in\mathbb{F}_{p^n}} \zeta_p^{-\mathrm{Tr}_m(bcG(z))}\zeta_p^{\mathrm{Tr}_n(za)}\zeta_p^{\mathrm{Tr}_m(bF(w))}\zeta_p^{-\mathrm{Tr}_n(wa)}\zeta^{\mathrm{Tr}_n(ua)}\\
=&p^{-n}\sum_{z,w\in\mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(w)-cG(z))}\sum_{a\in\mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_n((u+z-w)a)}\\
=&\sum_{z\in\mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(u+z)-cG(z))} = {}_c\mathfrak{C}_{F,G}(u,b).
\end{aligned}$$

The case when $F = G$ follows immediately using the same argument, and the claim is shown. ∎

Now, we can show the equivalence between perfect c-nonlinearity and c-differential bentness.

Theorem 3.3.3 *Let $1 \le m \le n$ be integers, p prime, F be an (n,m,p)-function, $1 \ne c \in \mathbb{F}_{p^m}$. Then F is perfect c-nonlinear if and only if F is c-differential bent. Moreover, F is strictly perfect c-nonlinear if and only if $W_F(a,b)\overline{W_F(a,bc)} = 0$, for all $a \in \mathbb{F}_{p^n}, b \in \mathbb{F}_{p^m}^*$.*

Proof: ($\Rightarrow$) We first assume that F is perfect c-nonlinear, and so, ${}_c\mathfrak{C}_F(u,b) = 0$, for all

$u \in \mathbb{F}_{p^n}^*$ and $b \in \mathbb{F}_{p^m}^*$. From Lemma 3.3.2, for an arbitrary $b \in \mathbb{F}_{p^m}^*$, we compute

$$\begin{aligned} W_F(a,b)\overline{W_F(a,bc)} &= \sum_{u \in \mathbb{F}_{p^n}} {}_c\mathfrak{C}_F(u,b)\zeta_p^{-\mathrm{Tr}_n(ua)} \\ &= {}_c\mathfrak{C}_F(0,b) + \sum_{0 \neq u \in \mathbb{F}_{p^n}} \zeta_p^{-\mathrm{Tr}_n(ua)} {}_c\mathfrak{C}_F(u,b) \\ &= {}_c\mathfrak{C}_F(0,b), \end{aligned}$$

where we use the assumption that the c-autocorrelations ${}_c\mathfrak{C}_F(u,b)$ are zero, except, possibly, at $u = 0$.

($\Leftarrow$) For the reciprocal, we assume that F is c-differential bent, that is, $W_F(a,b)\overline{W_F(a,bc)} = {}_c\mathfrak{C}_F(0,b)$, $b \neq 0$. Then, for any $b \in \mathbb{F}_{p^m}^*$ and $u \in \mathbb{F}_{p^n}^*$,

$$\begin{aligned} {}_c\mathfrak{C}_F(u,b) &= p^{-n} \sum_{a \in \mathbb{F}_{p^n}} W_F(a,b)\overline{W_F(a,bc)}\zeta_p^{\mathrm{Tr}_n(ua)} \\ &= p^{-n} {}_c\mathfrak{C}_F(0,b) \sum_{a \in \mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_n(ua)} = 0, \end{aligned}$$

where we use the property that the exponential sum of a balanced function (in this case $\mathrm{Tr}_n(ua)$, for $u \neq 0$) is zero. This proves the first claim, and we now show our second claim.

($\Rightarrow$) We assume that F is strictly perfect c-nonlinear. Using the same equations as above, but with the added fact that ${}_c\mathfrak{C}_F(u,b) = 0$ for all $u \in \mathbb{F}_{p^n}$, we arrive at

$$W_F(a,b)\overline{W_F(a,bc)} = {}_c\mathfrak{C}_F(0,b) = 0.$$

($\Leftarrow$) Again using the equations above, but this time with the fact that $W_F(a,b)\overline{W_F(a,bc)} = 0$ for all $a \in \mathbb{F}_{p^n}$, $b \in \mathbb{F}_{p^m}^*$, we see that ${}_c\mathfrak{C}_F(u,b) = 0$ for all $u \in \mathbb{F}_{p^n}$, $b \in \mathbb{F}_{p^m}^*$ and F is strictly perfect c-nonlinear. This completes the proof. ■

Earlier, we showed PcN is equivalent to balanced c-derivatives for $n = m$ or $p = 2$. Returning to the notion of perfect nonlinearity being the consequence of balanced derivatives, we now show a more general result for any p and with m not necessarily equal to n involving the traces of the c-derivatives.

Theorem 3.3.4 *Let F be an (n, m, p)-function, and $c \in \mathbb{F}_{p^m}$ fixed. Then F is a perfect c-nonlinear function (c-differential bent) if and only if, for all $b \neq 0, u \neq 0$ fixed, $x \mapsto \mathrm{Tr}_m(b(F(x+u) - cF(x))$ is balanced.*

Proof: ($\Rightarrow$) With $c \in \mathbb{F}_{p^n}$ constant, for every $u \in \mathbb{F}_{p^n}, b \in \mathbb{F}_{p^m}$, $0 \leq j \leq p-1$, we partition the traces of the differentials by letting $S_{j,c}^{u,b} = \{x \in \mathbb{F}_{p^n} \mid \mathrm{Tr}_m(b(F(x+u) - cF(x))) = j\}$.

Our goal is to show these partitions are all of the same size, and thus the traces of the derivatives are balanced. Now, consider the cyclotomic polynomial $\phi_p(x) = 1 + x + x^2 + \cdots + x^{p-1}$. Evaluating at ζ_p, which is a root of the cyclotomic polynomial, we deduce that $\zeta_p^{p-1} = -(1 + \zeta_p + \cdots + \zeta_p^{p-2})$. If $u \in \mathbb{F}_{p^n}^*, b \in \mathbb{F}_{p^m}^*$, and F is perfect c-nonlinear, then using our partition we obtain

$$0 = {}_c\mathfrak{C}_F(u, b) = \sum_{x \in \mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(x+u) - cF(x)))} = \sum_{j=0}^{p-1} |S_{j,c}^{u,b}| \zeta_p^j.$$

Now, using the identity for ζ_p^{p-1}, we have

$$\begin{aligned}
\sum_{j=0}^{p-1} |S_{j,c}^{u,b}| \zeta_p^j &= \left(\sum_{j=0}^{p-2} |S_{j,c}^{u,b}| \zeta_p^j \right) + |S_{p-1,c}^{u,b}| \zeta_p^{p-1} \\
&= \left(\sum_{j=0}^{p-2} |S_{j,c}^{u,b}| \zeta_p^j \right) - |S_{p-1,c}^{u,b}| (1 + \zeta_p + \cdots + \zeta_p^{p-2}) \\
&= \sum_{j=0}^{p-2} \left(|S_{j,c}^{u,b}| - |S_{p-1,c}^{u,b}| \right) \zeta_p^j.
\end{aligned}$$

To show the coefficients of ζ_p^j must be zero, we consider the field extension $\mathbb{Q} \overset{p-1}{\hookrightarrow} \mathbb{Q}(\zeta_p)$. This extension has degree $p-1$ and the elements in the set $\left\{ \zeta_p^j \mid 0 \leq j \leq p-2 \right\}$ are linearly independent in $\mathbb{Q}(\zeta_p)$ over $\mathbb{Q}$. Therefore, $|S_{j,c}^{u,b}| = |S_{p-1,c}^{u,b}|$ for all $0 \leq j \leq p-2$. In summary, the cardinality of the set $S_{j,c}^{u,b}$ is independent of j, and so, for all $c, b, u \neq 0$ fixed, the partitions of $\mathrm{Tr}_m(b(F(x+u) - cF(x))$ are of equal size and the traces of the derivatives are balanced.

($\Leftarrow$) If $x \mapsto \mathrm{Tr}_m(b(F(x+u) - cF(x))$ is balanced, then

$$ {}_c\mathfrak{C}_F(u,b) = \sum_{x \in \mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(x+u)-cF(x)))} = 0,$$

where we again use the property that the exponential sum of a balanced function (in this case the traces of the derivatives) is 0, and thus F is perfect c-nonlinear. ■

With this result, we can characterize the 0-differential bent functions.

Corollary 3.3.5 *Let F be an (n, m, p)-function. The following statements are equivalent:*

- (i) *F is a 0-differential bent (perfect 0-nonlinear) function;*
- (ii) *$W_F(0, b) = 0$, for all $b \neq 0$;*
- (iii) *(Under $m = n$) F is a permutation polynomial.*

Proof: If (i) holds, then by the previous theorem, the traces of the differentials of F are balanced when $c = 0$. That is, $\mathrm{Tr}_m(b(F(x+u))$ is balanced. Because $x + u$ is a bijection of input set $\mathbb{F}_{p^n}$, $\mathrm{Tr}_m(b(F(x+u))$ is balanced if and only if $x \mapsto \mathrm{Tr}_m(b(F(x))$ is balanced. Therefore, $W_F(0, b) = \sum_{x \in \mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(bF(x))} = 0$ and we have (ii). With (ii) holding and under $m = n$, by Lemma 3.2.2 F is a permutation polynomial and (iii) is established. Finally, again using [26, Theorem 7.7] as in Lemma 3.2.2, a permutation polynomial implies balanced derivatives when $c = 0$ and so F is 0-differential bent, and (iii) implies (i). ■

Thus, if $m = n$ and F is a permutation of $\mathbb{F}_{p^n}$, then F is 0-differential bent and perfect 0-nonlinear. For examples of c-differential bent (thus PcN) functions for all $c \neq 1$, consider $F(x) = x^{p^k}$, the linearized monomials on $\mathbb{F}_{p^n}$.

Corollary 3.3.6 *Let $F : \mathbb{F}_{p^n} \to \mathbb{F}_{p^n}$ be defined as $F(x) = x^{p^k}$. Then F is c-differential bent (perfect c-nonlinear) for all $c \neq 1$.*

Proof: Using the trace properties $\mathrm{Tr}_n(\alpha + \beta) = \mathrm{Tr}_n(\alpha) + \mathrm{Tr}_n(\beta)$ and $\mathrm{Tr}_n(\alpha^p) = \mathrm{Tr}_n(\alpha)$, we show these functions are PcN by computing the traces of their derivatives as

$$\mathrm{Tr}_n(b\,({}_cD_aF(x))) = \mathrm{Tr}_n\left(b((x+a)^{p^k} - cx^{p^k})\right)$$

$$
\begin{aligned}
&= \mathrm{Tr}_n\left(b(x^{p^k} + a^{p^k} - cx^{p^k})\right) \\
&= \mathrm{Tr}_n\left(b(1-c)x^{p^k}\right) + \mathrm{Tr}_n(ba) \\
&= \mathrm{Tr}_n\left((b(1-c))^{p^{-k}}x)^{p^k}\right) + \mathrm{Tr}_n(ba) \\
&= \mathrm{Tr}_n\left(b(1-c)^{p^{-k}}x\right) + \mathrm{Tr}_n(ba),
\end{aligned}
$$

which is balanced if $c \neq 1$ because the trace function of x times a constant is always balanced. ■

Thus, any linearized monomial is a (strictly) perfect c-nonlinear function for all $c \neq 1$. In fact, given any linearized polynomial L, for which $\mathrm{Tr}_n\left((1-c)^{p^{-k}}L(x)\right)$ is balanced, a similar argument shows that L is a (strictly) perfect c-nonlinear function for all $c \neq 1$. Furthermore, when $c = 0$ we can see this class of polynomials is a superclass of permutation polynomials.

3.4 Some Examples of PcN Functions

We provide next some examples demonstrating Corollaries 3.3.5 and 3.3.6. Using the open-source mathematical software system SageMath, we computed c-autocorrelation tables to exhibit several results of these Corollaries.

Example 3.4.1 *Consider $F(x) = x^5$ over $\mathbb{F}_{2^3}$, with $\mathbb{F}_{2^3} = \mathbb{F}_2[x]/\langle x^3 + x + 1\rangle$ and "a" as a root of $x^3 + x + 1$. As a permutation polynomial, by Corollary 3.3.5, F is 0-differential bent and hence 0-perfect nonlinear. Table 3.1 is the 0-autocorrelation table and confirms this is indeed true.*

$\boldsymbol{u\backslash b}$	0	1	a	$a+1$	a^2	a^2+1	a^2+a	a^2+a+1
0	8	0	0	0	0	0	0	0
1	8	0	0	0	0	0	0	0
a	8	0	0	0	0	0	0	0
$a+1$	8	0	0	0	0	0	0	0
a^2	8	0	0	0	0	0	0	0
a^2+1	8	0	0	0	0	0	0	0
a^2+a	8	0	0	0	0	0	0	0
a^2+a+1	8	0	0	0	0	0	0	0

Table 3.1. 0-Autocorrelation Table of F from Example 3.4.1

Example 3.4.2 *Consider $F(x) = x^4$ over $\mathbb{F}_{2^3}$ with $\mathbb{F}_{2^3}$ defined as in our previous example. If $p = 2$ and $k = 2$, then we have $x^4 = x^{p^k}$ and by Corollary 3.3.6, $F(x)$ is perfect c-nonlinear for all $c \neq 1$. The c-autocorrelation tables for $c \neq 1$ all match Table 3.1, confirming that $F(x)$ is PcN and c-differential bent for all $c \neq 1$.*

Other computations reveal more examples of c-different bent (perfect c-nonlinear) functions outside of the previous corollaries.

Example 3.4.3 *$F(x) = x^5$ over $\mathbb{F}_{3^3}$ is 2-differential bent. Defining $\mathbb{F}_{3^3}$ by $\mathbb{F}_3[x]/\langle x^3+2x+1\rangle$, we compute the 2-autocorrelation table of F. The result is a 27 by 27 matrix with 0's in all positions except for the trivial $b = 0$ column where the values are $p^n = 27$.*

Table 3.2 captures the previous examples and few others computed using SageMath.

$F(x)$	over	c-differential bent
x^5	$\mathbb{F}_{2^3}$	$c = 0$
x^4	$\mathbb{F}_{2^3}$	all $c \neq 1$
x^5	$\mathbb{F}_{3^3}$	$c = 0, 2$
x^{21}	$\mathbb{F}_{3^4}$	all $c \neq 1$
x^{15}	$\mathbb{F}_{3^3}$	$c = 0, 2$

Table 3.2. Examples of c-Differential Bent Functions

3.5 c-Avalanche Characteristics

As discussed in Subsection 2.3.3, the Boolean function property most closely related to Shannon's diffusion principle is the propagation criterion (PC), or more generally speaking, the avalanche characteristics. Recall a function is PC(k) if all the derivatives in the direction of vectors $\boldsymbol{a}$ with $1 \leq wt(\boldsymbol{a}) \leq k$ are balanced. As pointed out in Lemma 2.3.7, this is equivalent to the autocorrelation being 0 at those vectors. The definition was extended in Subsection 2.4.2 to vectorial BFs. When working in a finite field $\mathbb{F}$, the p-ary weight of an element $a \in \mathbb{F}$ is simply the number of non-zero positions in the vector representing a in the mapping between the finite field and the vector space. Now, equipped with the c-derivative and the c-autocorrelation function, we can generalize PC in a new direction.

Definition 3.5.1 *Let F be an (n, m, p)-function, $a \in \mathbb{F}_{p^n}$, $c \in \mathbb{F}_{p^m}$ and let $wt(a)$ be the weight of the p-ary vector representing a. Then F satisfies the* c-strict avalanche criterion *if, for all a such that $wt(a) = 1$, the derivative ${}_cD_aF(x)$ is balanced.*

Definition 3.5.2 *Let F be an (n, m, p)-function, $a \in \mathbb{F}_{p^n}$, $c \in \mathbb{F}_{p^m}$ and let $wt(a)$ be the weight of the p-ary vector representing a. Then F satisfies the* c-propagation criterion *of order k if, for all a such that $1 \leq wt(a) \leq k$, the derivative ${}_cD_aF(x)$ is balanced.*

Consider a function F that satisfies c-PC at input a and its component functions. Because the c-PC property is satisfied, the c-derivative in the direction of a is balanced. The absolute

trace of a balanced function is balanced, and therefore $\mathrm{Tr}_m(b(F(x+a)-cF(x)))$ is balanced. Thus we see that the probability that $\mathrm{Tr}_m(b(F(x+a)-cF(x))) = d$ is $\frac{1}{p}$ for all $d \in \mathbb{F}_p$ and we maintain the equivalent notions of balanced derivatives with probabilities of changing outputs in the original spirit of strict avalanche and propagation criterion.

The connection between PC, SAC and autocorrelation in Lemma 2.3.7 also holds for c-PC of (n, m, p)-functions. This is due to the same argument for made after Definition 3.2.1 for perfect c-nonlinear functions and the 0 autocorrelation values that are the result of balanced derivatives. We capture the same idea here with another lemma.

Lemma 3.5.3 *Let F be (n, m, p)-function and $wt(a)$ be the weight of the p-ary vector representing a. If F is c-PC(k), then the c-autocorrelation of F at a, $b \neq 0$, ${}_c\mathfrak{C}_F(a, b) = \sum_{x \in \mathbb{F}_{p^n}} \zeta_p^{\mathrm{Tr}_m(b(F(x+a)-cF(x)))} = 0$ for all a such that $1 \leq wt(a) \leq k$.*

A similar generalization and a thorough analysis of the strict avalanche criterion based on the c-differential was done in [29]. However, SAC (and c-SAC) is considered to be a local property of a function in that it provides a good indication of behavior when the input is changed in one position (i.e., by vectors of weight one). PC of order k, while stronger than SAC, is still a local property, providing information on how a function behaves when the input is changed by specific amounts. Unfortunately, a function that satisfies SAC or PC(k) may still have an undesirable linear structure of weight larger than k. In order to provide a better understanding of the overall behavior of a function under the action of changing the input, the authors of [30] introduced the so-called *Global Avalanche Characteristics (GAC)*.

In [30], two different measurements are defined to capture the GAC of a function: the *absolute* indicator and the *sum of squares* indicator. The absolute indicator is the largest absolute value of all non-trivial autocorrelation coefficients of a function, while the sum of squares indicator is the sum of all of the autocorrelation coefficients squared. The lower the values of both indicators, the better the GAC of the function. For instance, if the absolute indicator of a binary function is 2^n, the function has a linear structure, even if it has a good PC(k) properties for some k less than n. We will introduce the generalization of GAC to c-GAC in Section 3.5.2, but before that we consider the extension of GAC from classical Boolean to p-ary and (n, m, p)-functions, which may have been described before but we could not locate.

In the case of p-ary functions, for the absolute indicator we switch from the normal absolute value to the complex absolute value. For the sum of squares indicator, we sum the squares of the complex absolute values of the autocorrelation coefficients. We formally define these concepts, discuss some bounds, and then provide an example. Notice we return to traditional vector space definitions since we are not working with the multiplicative differential. We will return to finite field definitions when we explore c-GAC.

Definition 3.5.4 *Let f be a p-ary function, and $\mathfrak{C}_f(\boldsymbol{u})$ be the autocorrelation of f at $\boldsymbol{u} \in \mathbb{F}_p^n$. The* sum of squares indicator *for f, σ_f, is*

$$\sigma_f = \sum_{\boldsymbol{u} \in \mathbb{F}_p^n} \left| \mathfrak{C}_f(\boldsymbol{u}) \right|^2, \tag{3.4}$$

where $|\cdot|$ is the complex absolute value.

Definition 3.5.5 *Let f be a p-ary function, and $\mathfrak{C}_f(\boldsymbol{u})$ be the autocorrelation of f at $\mathbf{0} \neq \boldsymbol{u} \in \mathbb{F}_p^n$. The* absolute indicator *for f, ω_f, is*

$$\omega_f = \max_{\substack{\boldsymbol{u} \in \mathbb{F}_p^n \\ \boldsymbol{u} \neq 0}} \left| \mathfrak{C}_f(\boldsymbol{u}) \right|. \tag{3.5}$$

We can bound the indicators in a similar manner as done in [30].

Theorem 3.5.6 *Let f be a p-ary function. Then*

(i) $p^{2n} \leq \sigma_f \leq p^{3n}$;
(ii) $\sigma_f = p^{2n}$ *if and only if f is perfect nonlinear;*
(iii) $\sigma_f = p^{3n}$ *if and only if f is affine.*

Proof: For the lower bound in (i), we note that the autocorrelation of f is always p^n when $\boldsymbol{u} = 0$. Therefore, the minimum is achieved when the rest of the values of the autocorrelation spectrum are 0 and we have

$$\sigma_f \geq \left(\mathfrak{C}_f(\mathbf{0}) \right)^2 = (p^n)^2 = p^{2n}. \tag{3.6}$$

These functions with non-trivial autocorrelation spectrum of 0 are exactly the perfect nonlinear p-ary functions, and we have (ii). For the upper bound in (i), we note that the autocorrelation spectrum is maximum when every $\boldsymbol{u}$ is a linear structure of F and $\mathfrak{C}_f(\boldsymbol{u}) = p^n$. Thus, if all $\boldsymbol{u} \in \mathbb{F}_p^n$ are linear structures, we have

$$\sigma_f \leq p^n \left(\mathfrak{C}_f(u)\right)^2 = p^n (p^n)^2 = p^{3n}. \tag{3.7}$$

Functions whose linear structures are every element of $\mathbb{F}_p^n$ are affine functions, and we have (iii). ■

Corollary 3.5.1 *Let f be a p-ary function. Then*

(i) $0 \leq \omega_f \leq p^n$;
(ii) $\omega_f = 0$ *if and only if f is perfect nonlinear;*
(iii) $\omega_f = p^n$ *if and only if f has a linear structure.*

Corollary 3.5.1 follows the same argument as Theorem 3.5.6. We now provide an example.

Example 3.5.1 *Let $f : \mathbb{F}_3^2 \to \mathbb{F}_3$ have an ANF of $x_1^2 + x_2$ with the following truth table:*

x_2	x_1	f
0	0	0
0	1	1
0	2	1
1	0	1
1	1	2
1	2	2
2	0	2
2	1	0
2	2	0

Table 3.3. Truth Table of a p-ary Function f

The autocorrelation of f at $\boldsymbol{u}$ is complex valued function $\sum_{x\in\mathbb{F}_3^2} e^{(\frac{2\pi i}{3})^{(f(\boldsymbol{x}+\boldsymbol{u})-f(\boldsymbol{x}))}}$. Computing the coefficients using the definition, we have

$$
\begin{aligned}
\mathfrak{C}_f(00) &= 9(e^{(\frac{2\pi i}{3})})^0 = 9,\\
\mathfrak{C}_f(01) &= 6(e^{(\frac{2\pi i}{3})})^0 + 3(e^{(\frac{2\pi i}{3})})^1 = \frac{9}{2} + \frac{3\sqrt{3}}{2}i\,,\\
\mathfrak{C}_f(02) &= 3(e^{(\frac{2\pi i}{3})})^0 + 6(e^{(\frac{2\pi i}{3})})^1 = 3\sqrt{3}i\,,\\
\mathfrak{C}_f(10) &= 9(e^{(\frac{2\pi i}{3})})^1 = -\frac{9}{2} + \frac{9\sqrt{3}}{2}i\,,\\
\mathfrak{C}_f(11) &= 6(e^{(\frac{2\pi i}{3})})^1 + 3(e^{(\frac{2\pi i}{3})})^2 = -\frac{9}{2} + \frac{3\sqrt{3}}{2}i\,,\\
\mathfrak{C}_f(12) &= 3(e^{(\frac{2\pi i}{3})})^1 + 6(e^{(\frac{2\pi i}{3})})^2 = -\frac{9}{2} - \frac{3\sqrt{3}}{2}i\,,\\
\mathfrak{C}_f(20) &= 9(e^{(\frac{2\pi i}{3})})^2 = -\frac{9}{2} - \frac{9\sqrt{3}}{2}i\,,\\
\mathfrak{C}_f(21) &= 3(e^{(\frac{2\pi i}{3})})^0 + 6(e^{(\frac{2\pi i}{3})})^2 = 3\sqrt{3}i\,,\\
\mathfrak{C}_f(22) &= 6(e^{(\frac{2\pi i}{3})})^0 + 3(e^{(\frac{2\pi i}{3})})^2 = \frac{9}{2} - \frac{3\sqrt{3}}{2}i\,.
\end{aligned}
$$

We need the complex absolute values of the coefficients before we can calculate f's GAC. These values are captured in Table 3.4. Recall the trivial coefficient at $\boldsymbol{u} = \mathbf{0}$ is included in the sum of squares, but not the absolute indicator.

u	$\lvert\mathfrak{C}_f(u)\rvert$
00	9
01	$3\sqrt{3}$
02	$3\sqrt{3}$
10	9
11	$3\sqrt{3}$
12	$3\sqrt{3}$
20	9
21	$3\sqrt{3}$
22	$3\sqrt{3}$

Table 3.4. Complex Absolute Values of a p-ary Function's Autocorrelation

We see from the table that f has two linear structures, namely (10) *and* (20)*. Thus, the absolute indicator of f is $9 = p^n$. For the sum of squares indicator, we simply square the values in the table and sum.*

Similar to the other properties explored, moving from the single output to the multi-output functions we simply consider the autocorrelation coefficients of the component functions. Thus, for GAC of a (n, m, p)-function F, the absolute indicator, ω_F, is the largest absolute indicator of the component functions.

Definition 3.5.7 *Let F be an (n, m, p)-function, and $\mathfrak{C}_F(\boldsymbol{u}, \boldsymbol{b})$ be the autocorrelation of F at $\boldsymbol{u} \in \mathbb{F}_p^n$, $\boldsymbol{b} \in \mathbb{F}_p^m$. The* absolute indicator *for F, ω_F, is*

$$\omega_F = \max_{\substack{\boldsymbol{u}\in\mathbb{F}_p^n, \boldsymbol{u}\neq 0 \\ \boldsymbol{b}\in\mathbb{F}_p^m, \boldsymbol{b}\neq 0}} \left|\mathfrak{C}_F(\boldsymbol{u}, \boldsymbol{b})\right|. \tag{3.8}$$

For the sum of squares indicator, there are a few options. The sum could include the entire autocorrelation table, which contains the autocorrelation of each component function, or it

could be defined as the largest sum of squares of any single component function. We choose the latter, recalling that $\boldsymbol{b} \cdot F$ is not a component function when $\boldsymbol{b} = \mathbf{0}$.

Definition 3.5.8 *Let F be an (n, m, p)-function, and $\mathfrak{C}_F(\boldsymbol{u}, \boldsymbol{b})$ be the autocorrelation of F at $\boldsymbol{u} \in \mathbb{F}_p^n$, $\boldsymbol{b} \in \mathbb{F}_p^m$. The* sum of squares indicator *for F, σ_F, is*

$$\sigma_F = \max_{\substack{\boldsymbol{b} \in \mathbb{F}_p^m \\ \boldsymbol{b} \neq 0}} \Big(\sum_{\boldsymbol{u} \in \mathbb{F}_p^n, \boldsymbol{b} \in \mathbb{F}_p^m} \Big| \mathfrak{C}_F(\boldsymbol{u}, \boldsymbol{b}) \Big|^2 \Big). \tag{3.9}$$

The component functions of any (n, m, p)-function are p-ary functions, and thus the bounds remain the same. We need to explore one more concept before generalizing GAC with the c-derivative.

3.5.1 c-Linear Structures

Sections 3.2 and 3.3 discussed the optimal performance of a function under the c-differential cryptographic properties, namely perfect c-nonlinearity and c-differential bentness. Next, we consider the opposite. That is, the worst possible performance of a function's c-derivatives. In the case of $c = 1$, we know if derivatives are constant in some direction a, that a is a linear structure and represents a weakness. Even if other properties of the function are good (e.g., high nonlinearity, algebraic degree), an undesirable linear structure may still be present. If all $a \in \mathbb{F}_{p^n}$ are linear structures of a function, then the function is affine, the weakest class of cryptographic functions. Consider the equivalent notions for $c \neq 1$. That is, if F is an (n, m, p)-function and $F(x+a) - cF(x)$ is constant for all $x \in \mathbb{F}_{p^n}$, then we call a a *c-linear structure*. We now show that, while this definition of a c-linear structure as an extension of linear structures makes sense, when $c \neq 1$ there are no linear structures for non-constant (n, m, p)-functions when $p = 2$, or when $p > 2$ and $n = m$. In other words, if F is a non-constant univariate polynomial over a finite field, then it does not have a c-linear structure. Of course, constant functions are not used in cryptography as they do not depend on the input. We start by showing the case in binary.

Theorem 3.5.9 . *Let $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^m}$ be an (n, m)-function, with $a \in \mathbb{F}_{2^n}$, $1 \neq c \in \mathbb{F}_{2^m}$. Then the c-derivative of F, $F(x + a) - cF(x)$, is constant if and only if F is a constant function.*

In other words, only constant (n, m)-functions have c-linear structures when $c \neq 1$.

Proof: Let $F(x+a) - cF(x) = b$ for all $x \in \mathbb{F}_{2^n}$ for some $b \in \mathbb{F}_{2^m}$. Then $F(x+a) = b + cF(x)$. Now let $x :\longrightarrow x + a$. Then we also have $F(x) = b + cF(x + a)$. Substituting, we see

$$F(x) = b + c(b + cF(x)),$$

$$F(x) - c^2F(x) = b + cb,$$

$$F(x) = \frac{b(1 + c)}{1 - c^2} = \frac{b}{1 + c}.$$

Thus, F is a constant function. For the converse, if $F(x) = k$, then the c-derivative is $k - ck$, which is constant for fixed c. ■

In the more general case of odd characteristic we restrict the case to $n = m$, where we know there is a univariate polynomial representation for each function.

Theorem 3.5.10 *Let F be an (n, n, p)-function, with $a, c \in \mathbb{F}_{p^n}$ with $c \neq 1$. Then the c-derivative of F, $F(x + a) - cF(x)$, is constant if and only if F is a constant function. In other words, only constant (n, n, p)-functions have c-linear structures when $c \neq 1$.*

Proof: Let

$$F(x) = \sum_{i=0}^{t \leq p^n - 2} d_i x^i \ ,$$

where $d_i \in \mathbb{F}_{p^n}$ and $d_t \neq 0$ be the univariate polynomial representation of F. Then, if the c-derivative is constant, we have

$$F(x + a) - cF(x) = \sum_{i=0}^{t} d_i (x + a)^i - c \sum_{i=0}^{t} d_i x^i = b \ ,$$

with $b \in \mathbb{F}_{p^n}$. Expanding with the binomial theorem and moving b to the other side of the

equation, we arrive at

$$F(x+a) - cF(x) = \sum_{i=0}^{t} d_i \sum_{k=0}^{i} \binom{i}{k} x^k a^{i-k} - c \sum_{i=0}^{t} d_i x^i - b = 0 \, .$$

The degree of this polynomial is t, as the coefficient of x^t is $d_t(1-c)$, which is non-zero. Since a polynomial of degree t over a finite field can only have at most t roots, and $t \leq p^n - 2 < p^n$, we have fewer than p^n solutions for x. However, for a to be a linear structure, the polynomial must vanish (evaluate to 0) at all p^n values of x. Thus, we have shown that a is not a linear structure. For the converse, a constant function $F(x) = k$ has the c-derivative $k - ck$, which is constant and the theorem is shown. ■

We have demonstrated that c-linear structures in (n, m, p)-functions only arise in constant functions in even characteristic and when $n = m$ in odd characteristic. However, an S-box's component functions may still have weaknesses obscured by the lack of a c-linear structure from the finite field viewpoint. We can account for these with the notion of c-Global Avalanche Characteristics.

3.5.2 c-Global Avalanche Characteristics

With the c-autocorrelation, we return to finite fields and extend GAC by defining c-Global Avalanche Characteristics (c-GAC). As with our look at (n, m, p)-function GAC, we define the indicators based on the performance of the component functions.

Definition 3.5.11 *Let F be an (n, m, p)-function, $c \in \mathbb{F}_{p^m}$, and ${}_c\mathfrak{C}_F(u, b)$ be the c-autocorrelation of F at $u \in \mathbb{F}_{p^n}$, $b \in \mathbb{F}_{p^m}$. The* c-absolute indicator, *${}_c\omega_F$, for F is*

$${}_c\omega_F = \max_{\substack{u \in \mathbb{F}_{p^n} \\ b \in \mathbb{F}_{p^m}^*}} \left| {}_c\mathfrak{C}_F(u, b) \right|. \tag{3.10}$$

Definition 3.5.12 *Let F be an (n, m, p)-function, $c \in \mathbb{F}_{p^m}$, and ${}_c\mathfrak{C}_F(u, b)$ be the c-autocorrelation of F at $u \in \mathbb{F}_{p^n}$, $b \in \mathbb{F}_{p^m}$. The* c-sum of squares indicator, *${}_c\sigma_F$, for F*

is

$$ {}_c\sigma_F = \max_{b \in \mathbb{F}_{p^m}^*} \left(\sum_{\substack{u \in \mathbb{F}_{p^n} \\ b \in \mathbb{F}_{p^m}}} \left| {}_c\mathfrak{C}_F(u, b) \right|^2 \right). \tag{3.11} $$

Continuing in the spirit of [30] and our extension of GAC to p-ary and (n, m, p)-functions, we capture bounds and properties of c-GAC for $c \neq 1$.

Theorem 3.5.13 *Let F be an (n, m, p)-function and $c \neq 1$. Then*

(i) $p^{2n} \leq {}_c\sigma_F \leq p^{3n}$;
(ii) ${}_c\sigma_F = p^{2n}$ *if and only if F is perfect c-nonlinear;*
(iii) ${}_c\sigma_F = p^{3n}$ *if and only if F is constant.*

Proof: For the lower bound in (i), we note that the c-autocorrelation of F is always p^n when $b = 0$. Therefore, the minimum is achieved when the rest of the values of the c-autocorrelation spectrum are 0 and we have

$$ {}_c\sigma_F \geq ({}_c\mathfrak{C}_F(u, 0))^2 = (p^n)^2 = p^{2n}. \tag{3.12} $$

We previously classified these functions with non-trivial c-autocorrelation spectrum of 0 as perfect c-nonlinear, and we have (ii). For the upper bound in (i), we note that the c-autocorrelation spectrum is maximum when every u is a linear structure of F and ${}_c\mathfrak{C}_F(u, b) = p^n$. Thus, if all $u \in \mathbb{F}_{p^n}$ are linear structures, we have

$$ {}_c\sigma_F \leq p^n \, ({}_c\mathfrak{C}_F(u, b))^2 = p^n (p^n)^2 = p^{3n}. \tag{3.13} $$

We showed in Section 3.5.1 that only constant functions could have all c-linear structures, and we have (iii). ■

The following corollary for the c-absolute indicator follows immediately.

Corollary 3.5.2 *Let F be an (n, m, p)-function and $c \neq 1$. Then*

(*i*) $0 \leq {}_c\omega_F \leq p^n$;

(*ii*) ${}_c\omega_F = 0$ *if and only if* F *is perfect* c*-nonlinear;*

(*iii*) ${}_c\omega_F = p^n$ *if and only if* F *has a* c*-linear structure.*

When $c = 1$, we know the upper bounds of both indicators are determined by the performance of affine functions, because all elements of the base vector space or finite field are linear structures of affine functions. However, as we showed in Section 3.5.1, only constant (n, m, p)-functions have linear structures when $c \neq 1$. A natural question is how affine functions perform when $c \neq 1$. In the univariate polynomial representation of an affine function in a finite field, the powers of all monomials must p^k for some integer k. Consider the affine function $F(x) = dx^{p^k} + e$ over $\mathbb{F}_{p^n}$ with $d, e \in \mathbb{F}_{p^n}$. Using a similar argument as we did for Corollary 3.3.6, we have

$$\begin{aligned}
\mathrm{Tr}_n(b\ ({}_cD_aF(x))) &= \mathrm{Tr}_n\left(b(d(x+a)^{p^k} + e - c(dx^{p^k} + e))\right) \\
&= \mathrm{Tr}_n\left(b(dx^{p^k} + da^{p^k} + e - cdx^{p^k} - ce)\right) \\
&= \mathrm{Tr}_n\left(bd(1-c)x^{p^k}\right) + \mathrm{Tr}_n(bda) + \mathrm{Tr}_n(b(1-c)e) \\
&= \mathrm{Tr}_n\left((bd(1-c))^{p^{-k}}x)^{p^k}\right) + \mathrm{Tr}_n(bda) + \mathrm{Tr}_n(b(1-c)e) \\
&= \mathrm{Tr}_n\left(bd(1-c)^{p^{-k}}x\right) + \mathrm{Tr}_n(bda) + \mathrm{Tr}_n(b(1-c)e),
\end{aligned}$$

which is balanced by the same argument as Corollary 3.3.6. Therefore, affine functions of this type are PcN, for $c \neq 1$. We know affine functions are the weakest class of functions to use in a cryptographic transformation, and we arrive at a what may seem like a contradiction. However, we submit that when considering the performance of a cryptographic (n, m, p)-function, multiple properties must be considered simultaneously and *many* potential values of $c \in \mathbb{F}_{p^m}$ (including the obvious $c = 1$) may need to be considered. Good performance for some values c may not shield a function from the threat of a differential attack based on others.

3.6 Summary

In this chapter, we used the c-derivative introduced in [2] and a new autocorrelation function to generalize multiple cryptographic properties of (n, m, p)-functions. We extended

Nyberg's notion of perfect nonlinearity, developed the notion of c-differential bentness using a Walsh Transform product, and showed that the two are equivalent. Additionally, we generalized avalanche characteristics with p-ary and c-GAC definitions, providing a broader look into the diffusion properties of an S-box. In the next chapter, we continue our investigations with an analysis of how functions perform with another generalized cryptographic property, c-differential uniformity.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
c-Differential Uniformity and the Inverse Function

In Chapter 3 we used c-derivatives and a new c-autocorrelation to define the ideas of perfect c-nonlinearity, c-differential bentness, and c-avalanche characteristics, all of which are generalizations of the original notions. In [2], where the c-differential is first introduced as a potential expansion of differential cryptanalysis, the authors also defined the corresponding c-differential uniformity (cDU), generalizing the traditional notion of DU.

Definition 4.0.1 *Let F be an (n, n, p)-function with $a, b \in \mathbb{F}_{p^n}$, and let ${}_c\Delta_F(a, b) = |x \in \mathbb{F}_{p^n} : F(x + a) - cF(x) = b|$. The quantity ${}_c\delta_F = \max\{{}_c\Delta_F(a, b) : a, b \in \mathbb{F}_{p^n}, a \neq 0$ if $c = 1\}$ is called the* c-differential uniformity *of F.*

Maintaining the traditional equivalence of a PN function having a DU of 1, we see a PcN function (as introduced in Section 3.2) has a cDU of 1 because (n, n, p)-functions that are PcN have balanced c-derivatives (see Lemma 3.2.2). A function with a cDU of 2 is called *almost perfect c-nonlinear (APcN)*.

The authors of [2] also investigated known perfect nonlinear functions (recall PN functions do not exist when $p = 2$) to determine their behavior under the new cDU property. The introduction of the cDU concept sparked a large interest in the topic and multiple papers have since been submitted; including considerations of traditional APN functions, power functions, multinomials, constructions of functions with low cDU, and results on questions of existence of PcN and APcN functions under certain conditions. We compiled many of their findings into Section 4.4, providing a state-of-the-art on the topic.

The primary result of this chapter is the analysis of the behavior of the inverse function under certain extended affine (EA) equivalences in Section 4.2 and Section 4.3. Affine and EA transformations of the inverse function are common among S-boxes and we show that small perturbations can increase the cDU significantly.

4.1 The Multiplicative Inverse Function

The Advanced Encryption Standard (AES) became a symmetric block cipher standard of the United States federal government in 2001 after a lengthy open competition process [31]. AES is a subset of the more general *Rijndael* family of ciphers, created by and named after two Belgian cryptographers, Vincent Rijmen and Joan Daemen. The standard has proven to be very successful in both implementation and resistance against public cryptanalysis and is now in widespread use across the globe. Perhaps surprisingly, the only nonlinear component of AES, and several other popular ciphers, is an affine transformation of the so-called *inverse function* $F : \mathbb{F}_{p^n} \to \mathbb{F}_{p^n}$

$$F(x) = \begin{cases} \frac{1}{x} & \text{if } x \neq 0 \\ 0 & \text{if } x = 0 \end{cases}. \tag{4.1}$$

This function can be captured as a monomial in the form of $F(x) = x^{p^n-2}$. To see this, consider a finite field of order p^n (with multiplicative group of order $p^n - 1$). If $x = 0$, then obviously $F(x) = 0$. If $x \neq 0$, then, as a consequence of Lagrange's theorem [32, Theorem 10.10] we have $x^{p^n-1} = 1$. Thus $x^{p^n-2} = x^{-1}$ and $F(x)$ earns its name as the inverse function. In the AES S-box the inverse function is applied followed by an affine transformation. The composition of these actions results in a function that is affine equivalent to the inverse function. In the specific case of AES $p = 2$ and $n = 8$, but we treat the inverse function in the more general p^n setting in this chapter.

It is well known that differential uniformity is preserved under extended affine, and therefore, affine equivalence. As demonstrated in [33], c-differential uniformity holds under affine equivalences of the form $F \circ A$, where A is an affine transformation of the input. To see this, consider two functions F and G such that $G = F \circ A$ and consider the equation $G(x + a) - cG(x) = b$. This is the same as $(F \circ A)(x + a) - c(F \circ A)(x) = b$ or $F(A(x + a)) - cF(A(x)) = b$ where A is an affine mapping of $\mathbb{F}_{p^n}$. Since A is affine, the equation becomes $F(A(x) + A(a)) - cF(A(x)) = b$. Now, if we let $A(x) = y$ and $A(a) = \alpha$, then we have $F(y + \alpha) - cF(y) = b$. Because x, a, y, and α are in $\mathbb{F}_{p^n}$ and A is invertible, we can put solutions of $F(y + \alpha) - cF(y) = b$ in one-to-one correspondence with solutions to $G(x + a) - cG(x) = b$. Therefore, the equations have the same number of solutions and F and G have the same cDU. However, cDU does not hold under the more general affine

equivalence $B \circ F \circ A$ from Subsection 2.4.4 and therefore does not hold under general extended affine equivalence. This will be demonstrated in the following sections.

The differential uniformity of the inverse function over $\mathbb{F}_{2^n}$ is 2 for n odd and 4 for n even. In [2], the authors investigated the cDU of the inverse function for all prime characteristics. They proved the cDU of the inverse function in both even and odd characteristic is 1 for $c = 0$ (that is, the inverse function is a permutation, see Section 3.3), 2, or 3, based on the value of $c \neq 1$. In other words, the maximum cDU sometimes slightly decreases and sometimes slightly increases for the inverse function from the original notion of DU. While these changes are minimal in the case of the inverse function, later we will see in Section 4.4 that the cDU of other classes of functions can significantly increase from the original DU. Additionally, we will demonstrate in Chapter 6 that for other functions the maximum cDU is achieved only when $c = 1$ (i.e., traditional DU), and for $c \neq 1$ the differential count decreases. It will become apparent that the cDU property can behave in some unexpected ways. We demonstrate this in the next few sections by showing that small perturbations of the inverse function can significantly increase the cDU. The material in Section 4.2 and Section 4.3 is based on and expanded from Stănică and Geary [34].

4.1.1 Some Lemmas

The proofs of the theorems in the next section involve counting solutions of c-differential equations. We will make use of Hilbert's Theorem 90 and a few lemmas.

Theorem 4.1.1 Hilbert's Theorem 90, see [35].

Let $\mathbb{F} \hookrightarrow \mathbb{K}$ be a cyclic Galois extension and let σ be a generator of the Galois group $\mathrm{Gal}(\mathbb{K}/\mathbb{F})$. *Then for $x \in \mathbb{K}$, the relative trace* $\mathrm{Tr}_{\mathbb{K}/\mathbb{F}}(x) = \sum_{i=0}^{|\mathrm{Gal}(\mathbb{K}/\mathbb{F})|-1} \sigma^i(x) = 0$ *if and only if $x = \sigma(y) - y$, for some $y \in \mathbb{K}$.*

The field extension that we will be working in, $\mathbb{F}_p \hookrightarrow \mathbb{F}_{p^n}$, is a cyclic Galois extension with $\sigma : x \to x^p$ a generator of the Galois group. This will allow us to use Hilbert's Theorem 90 as needed. We will also need the following lemmas.

Lemma 4.1.2 *Let n be a positive integer. We have:*

(*i*) *The equation $ax^2 + bx + c = 0$, with $a, b, c \in \mathbb{F}_{2^n}$, $ab \neq 0$, has two solutions in $\mathbb{F}_{2^n}$ if $\mathrm{Tr}\left(\frac{ac}{b^2}\right) = 0$, and zero solutions, otherwise. If $a \neq 0, b = 0$, the solution is unique (see* [36]).

(*ii*) *The equation $ax^2+bx+c = 0$, with $0 \neq a, b \in \mathbb{F}_{p^n}$, p odd, has (two, respectively, one) solutions in $\mathbb{F}_{p^n}$ if and only if the discriminant $b^2 - 4ac$ is a (nonzero, respectively, zero) square in $\mathbb{F}_{p^n}$ (see* [2]).

(*iii*) (*see* [37]) *The equation $x^3 + ax + b = 0$, with $a, b \in \mathbb{F}_{2^n}$, $b \neq 0$, has (denoting by t_1, t_2 the roots of $t^2 + bt + a^3 = 0$):*

  (*i*) *three solutions in $\mathbb{F}_{2^n}$ if and only if $\mathrm{Tr}(a^3/b^2) = \mathrm{Tr}(1)$ and t_1, t_2 are cubes in $\mathbb{F}_{2^n}$ for n even, and in $\mathbb{F}_{2^{2n}}$ for n odd;*

  (*ii*) *a unique solution in $\mathbb{F}_{2^n}$ if and only if $\mathrm{Tr}(a^3/b^2) \neq \mathrm{Tr}(1)$;*

  (*iii*) *no solutions in $\mathbb{F}_{2^n}$ if and only if $\mathrm{Tr}(a^3/b^2) = \mathrm{Tr}(1)$ and t_1, t_2 are not cubes in $\mathbb{F}_{2^n}$ (n even), $\mathbb{F}_{2^{2n}}$ (n odd).*

Lemma 4.1.3 (*see* [2]*) Let p, t, n be integers greater than or equal to 1 (we take $t \leq n$, though the result can be shown in general). Then*

$$\gcd(2^t + 1, 2^n - 1) = \frac{2^{\gcd(2t,n)} - 1}{2^{\gcd(t,n)} - 1}, \text{ and if } p > 2, \text{ then,}$$
$$\gcd(p^t + 1, p^n - 1) = 2, \text{ if } \frac{n}{\gcd(n,t)} \text{ is odd,}$$
$$\gcd(p^t + 1, p^n - 1) = p^{\gcd(t,n)} + 1, \text{ if } \frac{n}{\gcd(n,t)} \text{ is even.}$$

Consequently, if either n is odd, or $n \equiv 2 \pmod 4$ and t is even, then $\gcd(2^t+1, 2^n-1) = 1$ and $\gcd(p^t + 1, p^n - 1) = 2$, if $p > 2$.

4.2 The c-Differential Uniformity of an EA-Perturbed Inverse Function

As described in Section 4.1, the inverse function has a very low cDU, ranging from 1 ($c = 0$), to 4 ($c = 1$, even n). In our main result of this chapter, we see that performing a simple modification of the inverse function significantly increases the maximum value in its

c-differential spectrum size. The modification we investigate is the addition of a linearized monomial of the form x^{p^t}. Recall that in characteristic p, $(a+b)^p = a^p + b^p$ by the identity often known as the "freshman identity" or "freshman mistake." Thus, if $f(x) = x^{p^t}$ over $\mathbb{F}_{p^n}$, then $f(x+y) = (x+y)^{p^t} = x^{p^t} + y^{p^t} = f(x) + f(y)$ and we see $f(x) = x^{p^t}$ is a linear function. Adding this linearized monomial to a function is an extended affine transformation and is the focus of our exploration in this section.

In the following, we take $n \geq 4$ an integer and $1 \leq t < n$ an integer such that $\gcd(t, n) = d$. We show the significant change in maximum cDU will occur when $a^{p^t+1} + 1 = 0$ has a root (and consequently, $\gcd(p^t+1, p^n-1)$ roots) in the field $\mathbb{F}_{p^n}$. This last condition will always happen if $p = 2$ or if n/d is even; this was shown in [38] using the fact that $\gcd(p^t+1, p^n-1)$ divides $\frac{p^n-1}{2}$ under these conditions.

Theorem 4.2.1 *Let p be a prime number, $n \geq 4$, $F(x) = x^{p^n-2}$ be the inverse function on $\mathbb{F}_{p^n}$, and $1 \neq c \in \mathbb{F}_{p^n}$. Then, the c-differential uniformity, ${}_c\delta_G$, of $G(x) = F(x) + x^{p^t}$ satisfies $p^{\gcd(n,t)} + 2 \leq {}_c\delta_G \leq p^t + 4$, if $p = 2$, or if $p > 2$ and $\frac{n}{\gcd(n,t)}$ is even; and $4 \leq {}_c\delta_G \leq p^t + 4$, if $p > 2$ and $\frac{n}{\gcd(n,t)}$ is odd.*

Proof: The c-differential uniformity equation of $G(x)$ for $c \in \mathbb{F}_{p^n}$ at $(a, b) \in \mathbb{F}_{p^n} \times \mathbb{F}_{p^n}$ is

$$(x+a)^{p^n-2} + (x+a)^{p^t} - c(x^{p^n-2} + x^{p^t}) = b. \tag{4.2}$$

We first assume that $a \neq 0$. We arrive at the upper bound by considering several cases.

Case (i). Let $x = 0$. Equation (4.2) becomes

$$\frac{1}{a} + a^{p^t} = b.$$

Thus, for any $a \neq 0$ and $b = \frac{1}{a} + a^{p^t}$, we have a solution of (4.2), for arbitrary c.

Case (ii). Let $x = -a$. Equation (4.2) becomes

$$c\left(\frac{1}{a} + a^{p^t}\right) = b,$$

and we have another solution to (4.2), for a given c by the above displayed equation. If a is such that $a^{p^t+1}+1=0$, then $\frac{1}{a}+a^{p^t}$ is 0 and therefore b must be zero. In this case c can be taken arbitrary. Note there are $\gcd(p^t+1, p^n-1)$ such a's (which, by Lemma 4.1.2, is $\gcd(2^t+1, 2^n-1) = \frac{2^{\gcd(2t,n)}-1}{2^{\gcd(t,n)}-1}$ if $p=2$, and if $p>2$, the number of such a's is 2 when $\frac{n}{\gcd(n,t)}$ is odd, and $p^{\gcd(n,t)}+1$, when $\frac{n}{\gcd(n,t)}$ is even, all if $t>0$; when $t=0$, the value of $\gcd(p^t+1, p^n-1)$ is 1, respectively, 2, for $p=2$, respectively, $p>2$).

We make an observation here: the two solutions from Cases (i) and (ii) cannot be combined for arbitrary c unless $b=0$ and $a^{p^t+1}+1=0$. Or, if $c=1$, then $b=\frac{1}{a}+a^{p^t}$.

Case (iii). Let $x \neq 0, -a$. Using the fact that any non-zero value raised to p^n-2 is its inverse and that x^{p^t} is linearized in $\mathbb{F}_{p^n}$, Equation (4.2) becomes

$$\begin{aligned}
&\frac{1}{x+a}+(1-c)x^{p^t}-\frac{c}{x}=b-a^{p^t}, \text{ that is,}\\
&x+(1-c)x^{p^t+1}(x+a)-c(x+a)=(b-a^{p^t})x(x+a), \text{ or,}\\
&x^{p^t+2}+ax^{p^t+1}+\frac{b-a^{p^t}}{c-1}x^2+\frac{ab+c-a^{p^t+1}-1}{c-1}x+\frac{ac}{c-1}=0. \qquad (4.3)
\end{aligned}$$

Therefore, the maximum number of solutions to (4.2) for $x \neq 0, -a$ is bounded by the degree of this polynomial, p^t+2. Combining with the previous two solutions from Cases (i) and (ii), we can bound the total number of solutions (and thus the the c-differential uniformity) by $2+p^t+2=p^t+4$. That is, ${}_c\delta_G \leq p^t+4$ and we have our claimed upper bound.

The most straightforward method to show the lower bound of ${}_c\delta_G$ is to consider the case of $a=b=0$. However, there are many other values of $(a,b) \in \mathbb{F}_{p^n} \times \mathbb{F}_{p^n}$ such that ${}_c\delta_G$ is lower bounded by $p^{\gcd(n,t)}+2$. We will start with the case of $a=b=0$.

When $a=0$, Equation (4.2) becomes

$$x^{p^n-2}+x^{p^t}=\frac{b}{1-c}.$$

If $b=0$, this equation has $x=0$ as a root, and moreover, we can factor into

$$x^{p^t}(x^{p^n-p^t-2}+1)=0.$$

Thus, when $x \neq 0$ we have $x^{p^n - p^t - 2} + 1 = 0$, and when multiplied through by x^{p^t+1} this is equivalent to $x^{p^t+1} + 1 = 0$. By [38], when n/d is even this always has a solution and by Lemma 4.1.3 there are $\gcd(p^t + 1, p^n - 1) = p^{\gcd(n,t)} + 1$ solutions. Along with the $x = 0$ solution, we arrive at the lower bound of $p^{\gcd(n,t)} + 2$ for n/d even.

We return to Case (iii) $(x \neq 0, -a)$, and proceed to establish the lower bound for $b \neq 0$ for $p > 2$. The case of $p = 2$ will be handled afterward. From Equation (4.3) we let $a = 0$, obtaining

$$x^{p^t+2} + \frac{b}{c-1}x^2 + x = 0, \tag{4.4}$$

with solution $x = 0$ and cofactor

$$x^{p^t+1} + \frac{b}{c-1}x + 1 = 0. \tag{4.5}$$

We now move to put (4.5) into a form of which there are known results on the number of solutions. First, relabel $x \mapsto \frac{1-c}{b}x$, for $b \neq 0$. Note if $b = 0$, we return to $x^{p^t+1} + 1 = 0$ and our previous argument.

Next, after relabeling (4.5), we multiply by $\left(\frac{b}{c-1}\right)^{p^t+1}$ and obtain

$$x^{p^t+1} - Bx + B = 0, \tag{4.6}$$

where $B = \left(\frac{b}{c-1}\right)^{p^t+1}$ and we can apply [39, Theorem 5.6]. Using the notations from [39], we let $\mathbb{F}_Q = \mathbb{F}_{p^n} \cap \mathbb{F}_{p^t} = \mathbb{F}_{p^{\gcd(n,t)}}$. Thus, $Q = p^{\gcd(n,t)}$. If we let $m = \frac{n}{\gcd(n,t)}$, [39, Theorem 5.6] tells us there are $\dfrac{Q^{m-1} - Q}{Q^2 - 1}$, $\dfrac{Q^{m-1} - 1}{Q^2 - 1}$, for m even, respectively, odd, values of B such that Equation (4.5) has $Q + 1 = p^{\gcd(n,t)} + 1$ solutions. We let T represent the set of B's that are these solutions. Thus, $|T| = \dfrac{Q^{m-1} - Q}{Q^2 - 1}$, for m even, and $|T| = \dfrac{Q^{m-1} - 1}{Q^2 - 1}$, for m odd.

To arrive at our claimed lower bound, we need to show we can find b, c such that B from (4.6) is in T. Start with $p > 2$ and m odd. Then by Lemma 4.1.3 we have $\gcd(p^t + 1, p^n - 1) = 2$. We define $\tilde{B}$ as the square roots of $B \in T$, and we have $b = (c-1)\tilde{B}^{\frac{2}{p^t+1}}$ for any $c \neq 1$. Such a $\tilde{B}$ always exists, for instance $\tilde{B} = 0$. The number of solutions of (4.5) for these

parameters is therefore $Q+1$. For m even, then by Lemma 4.1.3 $\gcd(p^t+1, p^n-1) = Q+1$. We again use [39], by taking $B = \tilde{B}^{Q+1} \in T$, and for a fixed c we have $b = (c-1)\tilde{B}^{\frac{Q+1}{p^t+1}}$. Again, such a $\tilde{B}$ always exists, for instance $\tilde{B} = 0$. The number of solutions of (4.5) for these parameters is therefore $Q+1 = p^{\gcd(n,t)}+1$. Along with the previous solution of $x = 0$, we have ${}_c\delta_G \geq p^{\gcd(n,t)}+2$. Although we cannot always guarantee a nonzero $\tilde{B}$, this argument shows there are many potential values of b, c that force the lower bound to $p^{\gcd(n,t)}+2$.

For $p = 2$, we use [40], where it was shown that an equation of the form $x^{2^t+1} + x + A = 0$ has $Q+1$ zeros for $\frac{Q^{m-1}-1}{Q^2-1}$, $\frac{Q^{m-1}-Q}{Q^2-1}$, for m odd, respectively, even, values of the parameter A. Next, we multiply (4.5) by $\left(\frac{b}{c+1}\right)^{\frac{1}{2^t}}$, which always exists because $\gcd(2^n-1, 2) = 1$ and therefore we can take repeated square roots. If we perform the substitution $x \mapsto x\left(\frac{b}{c+1}\right)^{\frac{1}{2^t}}$, we get the equation $x^{2^t+1} + x + \left(\frac{c+1}{b}\right)^{1+\frac{1}{2^t}} = 0$, and we can apply the same technique as in the previous proof, though, the existence of values b, c such that $A = \left(\frac{c+1}{b}\right)^{\frac{2^t+1}{2^t}}$ is not in question anymore for any $A \neq 0$. Therefore, when $p = 2$ and the conditions on n, t are met, we have ${}_c\delta_G \geq p^{\gcd(n,t)}+2$. The theorem is shown. ■

The following corollary is immediate and implies a significant increase in the maximum cDU of the inverse function after a particular EA transformation.

Corollary 4.2.2 *Let $n \geq 4$, $F(x) = x^{p^n-2}$ be the inverse function on $\mathbb{F}_{p^n}$, and $t \mid n$ be the largest divisor of n such that $\frac{n}{\gcd(n,t)}$ is even, and $G(x) = F(x) + x^{p^t}$. Then, there exists c such that ${}_c\delta_G \geq p^t + 2$.*

4.2.1 Particular Case of an EA Perturbed Inverse Function Over $\mathbb{F}_{2^8}$

We now use Corollary 4.2.2 to investigate the maximum cDU of a particular inverse function perturbed by a specific linearized monomial. We mentioned in Section 4.1 that in the case of AES the nonlinear component is an affine transformation of the inverse function with $p = 2$ and $n = 8$. We use this as motivation to choose the inverse function $F(x) = x^{2^8-2} = x^{254}$ over F_{2^8} to investigate.

Using Corollary 4.2.2 with $t = 4$, let $G(x) = x^{254} + x^{2^4}$. Then, we have $\frac{n}{\gcd(n,t)} = 2$, and

the conditions of the corollary are met, guaranteeing there exists some c such that ${}_c\delta_G$ is lower bounded by $p^{\gcd(n,t)} + 2 = 2^4 + 2 = 18$. In fact, we confirmed computationally using SageMath that ${}_c\delta_G = 18$. This maximum value occurs when $c = 0$. In other words, ${}_0\delta_G = 18$.

After computational methods provide us with the specific $c = 0$ that attains the maximum cDU, we can analytically pursue the results. With $G(x) = x^{254} + x^{2^4}$, the c-differential equation becomes

$$(x + a)^{254} + (x + a)^{16} - c(x^{254} + x^{16}) = b. \tag{4.7}$$

With $c = 0$, and if $x \neq a$, then we have

$$\begin{aligned}
&(x + a)^{254} + (x + a)^{16} - 0(x^{254} + x^{16}) = b,\\
&\frac{1}{x + a} + x^{16} + a^{16} = b,\\
&1 + x^{17} + ax^{16} + a^{16}x + a^{17} = bx + ba,\\
&x^{17} + ax^{16} + (a^{16} + b)x + (a^{17} + ba + 1) = 0.
\end{aligned}$$

For $b = 0$, this polynomial has 17 roots for many values of $a \in \mathbb{F}_{2^8}$. Some examples found in SageMath include α^2, α^4, where α is a primitive element in the finite field. When $x = a$, we simply have $0^{254} + 0^{16} = 0 = b$, providing one additional solution and resulting in 18 total.

Thus we reach our theoretical cDU of 18 on this particular inverse function perturbed by a specific linearized monomial. Recall that x^{254} over $\mathbb{F}_{2^8}$ has traditional DU of 4 and cDU values of 1,2, or 3 for all $c \neq 1$. This demonstrates that performing a small extended affine perturbation can cause a significant increase.

A similar analysis (both computationally and analytically) for $n = 10$ and $t = 5$ (respectively, $n = 12$ and $t = 6$) results in a maximum cDU of 34, (resp 66) achieved when $c = 0$.

These examples reinforce our key point of this section: unlike classical DU, cDU is not in general preserved by an extended affine transformation. Thus, if a real-world c-differential attack is ever realized, an S-box that uses a function with low cDU as a primitive cannot

be guaranteed to maintain low cDU after certain transformations. We compute the cDU performance of AES's actual S-box in Chapter 6.

4.2.2 Attaining the Upper or Lower Bounds on cDU

Next, we find some values of t for which the upper bound $p^t + 4$, or the lower bound $p^{\gcd(t,n)} + 2$ of Theorem 4.2.1 are attained by ${}_c\delta_G$ for some c. When $p = 2$, we will show that this happens for $t = 0$ and n even (upper) and n odd (lower).

Theorem 4.2.3 *Let $n \geq 4$, $F(x) = x^{2^n-2}$ be the inverse function on $\mathbb{F}_{2^n}$, and $1 \neq c \in \mathbb{F}_{2^n}$. Then, if n is even, the c-differential uniformity of $G(x) = F(x) + x$ is ${}_c\delta_G = 5$, for some c; if n is odd, there exists c such that ${}_c\delta_G = 4$. Moreover, if $G(x) = F(x) + x^2$ and n is even, then there exists c such that ${}_c\delta_G = 5$; if n is odd and there exists an a such that $\mathrm{Tr}\left(\frac{a^2}{a^2+a+1}\right) = \mathrm{Tr}\left(\frac{a^4}{(a+1)^5}\right) = 0$, then ${}_c\delta_G = 5$ for some c (for example, $c = 1 + \frac{1}{(a^3+a^2+1)^{\frac{1}{2}}}$).*

Proof: We will not go through the corresponding Cases (i) and (ii) as in Theorem 4.2.1 since these arguments are independent of t, but we will refer to them.

Let $G(x) = F(x) + x$ over $\mathbb{F}_{2^n}$. We must investigate the equation

$$(x + a)^{p^n-2} + (x + a) + c(x^{p^n-2} + x) = b. \tag{4.8}$$

Using the same rearrangement techniques that we used to arrive at Equation (4.3), with $t = 0$ we get

$$x^3 + \left(a + \frac{b + a}{1 + c}\right)x^2 + \frac{1 + c + ab + a^2}{1 + c}x + \frac{ac}{1 + c} = 0. \tag{4.9}$$

To achieve the maximum 5 number of solutions for x, we need 3 roots of this cubic combined with the two solutions from Cases (i) and (ii). That is, we need $a^{2^0+1} + 1 = b = 0$, which forces $a = 1$, $b = 0$, and the cubic becomes

$$x^3 + \frac{c}{1 + c}x^2 + \frac{c}{1 + c}x + \frac{c}{1 + c} = 0. \tag{4.10}$$

Next, we use the substitution $y = x + \frac{c}{c+1}$ in order to put the equation into a form where we

can use Lemma 4.1.2, arriving at

$$y^3 + \frac{c}{(c+1)^2}y + \frac{c}{(c+1)^2} = 0. \tag{4.11}$$

By Lemma 4.1.2, this last equation has three solutions if and only if $c \neq 0$ and $\mathrm{Tr}\left(\frac{c}{(c+1)^2}\right) = \mathrm{Tr}(1)$ and the roots t_1, t_2 of $t^2 + \frac{c}{(c+1)^2}t + \left(\frac{c}{(c+1)^2}\right)^3 = 0$ are cubes in $\mathbb{F}_{2^n}$ for n even, $\mathbb{F}_{2^{2n}}$ for n odd.

Starting with n even, we see that $\mathrm{Tr}\left(\frac{c}{(c+1)^2}\right) = \mathrm{Tr}\left(\frac{c+1+1}{(c+1)^2}\right) = \mathrm{Tr}\left(\frac{1}{c+1} + \frac{1}{(c+1)^2}\right)$. Using Hilbert's Theorem 90 with $\sigma : x \to x^2$ and $x = \frac{1}{c+1}$, we have $\mathrm{Tr}\left(\frac{1}{c+1} + \frac{1}{(c+1)^2}\right) = 0 = \mathrm{Tr}(1)$. Therefore, three solutions can only be potentially achieved if n is even.

Next we need to show we can always find some c, such that the solutions to $t^2 + \frac{c}{(c+1)^2}t + \left(\frac{c}{(c+1)^2}\right)^3 = 0$ are cubes in $\mathbb{F}_{2^n}$. The roots of this equation can be quickly found to be

$$t_1 = \frac{c}{(c+1)^3},\ t_2 = \frac{c^2}{(c+1)^3}.$$

If we take c to be a cube, then both of these roots are cubes, and consequently we have three roots for (4.10). Next, we need to argue that they are not repeated roots. Since we are working over binary, it is sufficient to check that the coefficient of x^2 in (4.10) , that is $\frac{c}{c+1}$, is not a root, which is true because the left hand side of (4.10) at $\frac{c}{c+1}$ is exactly $\frac{c}{(c+1)^2} \neq 0$, because $c \neq 0$. We therefore have three solutions, and with the two solutions from Cases (i) and (ii), we have our claimed five solutions when n is even.

For n odd, we cannot combine Cases (i) and (ii), but the same argument reveals four solutions for (4.10) and our claims for $x^{2^n-2} + x$ are shown.

Let now $t = 1$, that is, $G(x) = x^{2^n-2} + x^2$. If $b = 0$ Equation (4.3) becomes

$$x^4 + ax^3 + \frac{a^2}{1+c}x^2 + \frac{c + a^3 + 1}{1+c}x + \frac{ac}{1+c} = 0.$$

As in the previous argument for $t = 0$, we combine Cases (i) and (ii) to arrive at our count,

so we let $a = 1$. Note for n even we could also use $a^2 + a + 1 = 0$; for n odd, we can only have $a = 1$. When $a = 1$, the above equation becomes

$$x^4 + x^3 + \frac{1}{c+1}x^2 + \frac{c}{1+c}x + \frac{c}{1+c} = 0,$$

which when multiplied through by $1 + c$ becomes

$$(1+c)x^4 + (1+c)x^3 + x^2 + cx + c.$$

This can be factored as

$$(x^2 + x + 1)((c+1)x^2 + c) = 0.$$

For n even, we get 2 roots from the first factor and another from the second. Therefore, we get 3 roots for the above equation, which when combined with the 2 from Cases (i) and (ii) renders 5 altogether. There are many values of c we can take: for example, for any $x \neq 0, 1, a$ not a root of $x^2 + x + 1$, then we take $c = \frac{x^2}{x^2+1}$.

If n is odd, then $a = 1$ cannot give us more than 3 roots (when n is odd $x^2 + x + 1 \neq 0$, denying us 2 possible roots of the 5 with n even), so we assume that $a \neq 1$. Again, under n odd, if $b = 0$ and $c = 0$, Equation (4.3) becomes

$$x^4 + ax^3 + a^2x^2 + (1 + a^3)x = 0,$$

with solutions $x = 0, a + 1$, and $(x + a)^2 + (x + a) + 1 = 0$, but for n odd the last equation cannot hold. Next, we take $\frac{1}{a} + a^{2^t} = b$ (Case (i)), and with $b = 0$ we have $\frac{1}{a} = a^2$. Returning to Equation (4.3) with $\frac{1}{a} = a^2$, we see

$$x^4 + ax^3 + \frac{1}{a(c+1)}x^2 + \frac{c}{c+1}x + \frac{ac}{c+1} = 0.$$

Next, we find some values of a, c such that the above polynomial can be factored as

$$x^4 + ax^3 + \frac{1}{a(c+1)}x^2 + \frac{c}{c+1}x + \frac{ac}{c+1} = (x^2 + Ax + a)\left(x^2 + Bx + \frac{c}{c+1}\right).$$

Solving the obtained system, we find that

$$A = \frac{a^2(c+1)+c}{a(c+1)+c},\ B = \frac{(a+1)c}{a(c+1)+c},$$
$$\text{when } c = \left(\frac{a^3+a^2}{a^3+a^2+1}\right)^{1/2} = 1 + \frac{1}{(a^3+a^2+1)^{\frac{1}{2}}}.$$

Moreover, each factor in the factorization above has two distinct roots (when $AB \neq 0$) if $\mathrm{Tr}\left(\frac{a}{A^2}\right) = \mathrm{Tr}\left(\frac{a^2}{a^2+a+1}\right) = 0$ and $\mathrm{Tr}\left(\frac{c}{(c+1)B^2}\right) = \mathrm{Tr}\left(\frac{a^4}{(a+1)^5}\right) = 0$. Under the assumption that there are values of $a \neq 1$ for n odd such that both of these traces are 0 (computation reveals that it always happens, but we have been unable to show that in general), the claim is shown. ■

Thus, when $t = 0$ the upper bound determined by Theorem 4.2.1 of $2^0 + 4 = 5$ is achieved. When $t = 1$, we do not know if $2^1 + 4 = 6$ is achievable, but we showed there are multiple values of c such that, if $G(x) = x^{2^n-2} + x^2$, then ${}_c\delta_G = 5$ for n both even and odd.

Summarizing this section, the linearized monomial has an interesting impact on the cDU of the inverse function, causing a surprising increase under certain conditions. It is natural to consider how other transformations might affect the cDU. Next, we investigate a more general perturbation, a linearized polynomial.

4.3 EA Transformation of the Inverse Function by a Linearized Polynomial

The inverse function over finite fields is a popular block cipher primitive because it is a balanced function with high nonlinearity and low differential uniformity, among other good properties. As discussed previously in this chapter, the inverse function also performs well under the new c-differential uniformity property, but sees a significant increase with the addition of certain monomials. Here, we consider a natural continuation—how the inverse function performs under the more general EA transformations by linearized polynomials.

Consider polynomials of the form $L(x) = \sum_{i=0}^{n-1} a_i x^{p^i}$ over $\mathbb{F}_{p^n}$, with $a_i \in \mathbb{F}_{p^n}$. The same "freshman mistake" in characteristic p described at the beginning of Section 4.2 for a linearized monomial applies to these multinomials as well. Thus, if $G(x) = x^{p^n-2} + L(x)$,

we have another EA transformation of the inverse function.

Counting solutions to the c-differential equations over $\mathbb{F}_{p^n}$ that result from adding polynomials is a more challenging task than the monomials considered in Section 4.2. While the bounds we find are not as clean as in the case of the linearized monomial, we are able to use [41] to find some meaningful results. First, we introduce the results of [41] on the exponential sums of so-called *Dembowski–Ostrom*, or DO, polynomials.

In general, a DO polynomial, $D(x)$, is of the form $\sum_{i=1}^{n} a_i x^{p^{\alpha_i}+p^{\beta_i}}$. If we let $\alpha_i = i$ and $\beta_i = 0$, the polynomial becomes $\sum_{i=1}^{n} a_i x^{p^i+1}$. This is the form of polynomials we will encounter in the proof of the bounds on the cDU of $G(x)$.

Recall from Section 3.2 that $\chi_1(a) = \zeta_q^{\mathrm{Tr}_n(a)} = e^{\left(\frac{2\pi i \mathrm{Tr}_n(a)}{q}\right)}$ is the canonical additive character of $\mathbb{F}_q$, with $q = p^n$. A primary result of [41] tells us the number of solutions of the exponential sum of this polynomial is

$$\left| \sum_{x \in \mathbb{F}_q} \chi_1 \left(\sum_{i=0}^{n-1} a_i x^{p^i+1} \right) \right| = \sqrt{qN_\alpha}, \tag{4.12}$$

where N_α is the number of solutions for $T_n(w) = 0$, and

$$T_n(w) = 2A_0 w + \sum_{i=1}^{n-1} \left(A_i w^{p^i} + (A_i w)^{p^{n-i}} \right),$$

with $A_i = (\alpha a_i)^{p^{n-i}}$.

While it may not seem that much progress has been made in arriving at a new polynomial to solve, the author points out that T_n acts as linear operator over $\mathbb{F}_p$, and therefore finding the number of solutions is equivalent to relatively easy task of finding the rank of an n by n matrix. In fact, if $s_1, \ldots, s_k$ are the indices i where $a_i \neq 0$, and if $\epsilon = \gcd_{1 \leq i \leq n-1}\{2s_0, s_0 + s_i, s_0 + n - s_i, n\}$, then the number of solutions to $T_n(w) = 0$ is $p^{\epsilon\gamma_\alpha}$, for some nonnegative integer γ_α. We also note that character sums such as Equation (4.12) over finite fields are called *Weil* sums.

Before we proceed with our primary result of this section, we follow some of the notation

in [41] and list some technical conditions in that apply to our lower bound.

$$n = 2m; n/\delta \text{ is even}; 2\delta \mid s_i - s_j; 4 < p^\delta + 1 \mid p^{s_i} + 1. \tag{4.13}$$

Theorem 4.3.1 *Let p be an odd prime number, $n \geq 4$, $F(x) = x^{p^n-2}$ be the inverse function on $\mathbb{F}_{p^n}$, and $1 \neq c \in \mathbb{F}_{p^n}$. Let $L(x) = \sum_{i=0}^{n-1} a_i x^{p^i}$ be a linearized polynomial. Then, the c-differential uniformity, ${}_c\delta_G$, of $G(x) = F(x) + L(x)$ satisfies*

(i) ${}_c\delta_G \leq (pN)^{\frac{n}{2}}$, *where $N = \max_{\alpha \in \mathbb{F}_q}\{N_\alpha\}$, and N_α is the number of solutions w to*

$$T_n(w) = 2\alpha a_0 w + \sum_{i=1}^{n-1} \left((\alpha a_i)^{p^{-i}} w^{p^i} + (\alpha a_i)^{p^{-2i}} w^{p^{-i}} \right) = 0. \tag{4.14}$$

In fact, $N_\alpha = p^{\delta\gamma_\alpha}$, for some nonnegative integer γ_α.

(ii) ${}_c\delta_G \geq \dfrac{1}{p^n} \displaystyle\sum_{\alpha \in \mathbb{F}_q} \chi_1(\alpha)\mu_\alpha p^{\frac{\delta\gamma_\alpha}{2}}$, *under the conditions of* (4.13)*, where γ_α is defined in (i) and $\mu_\alpha = \pm 1$ is the sign of the Weil sum from Equation* (4.12)*. Even more precisely,*

$${}_c\delta_G \geq (-1)^{\frac{m}{\delta}} p^{-m} \sum_{\substack{\alpha \in \mathbb{F}_q \\ N_\alpha = 1}} \chi_1(\alpha) + (-1)^{\frac{m}{\delta}} p^{-m} \sum_{\substack{\alpha \in \mathbb{F}_q \\ N_\alpha > 1}} \chi_1(\alpha)(-1)^{\frac{\gamma_\alpha}{2}} p^{\frac{\delta\gamma_\alpha}{2}}.$$

Proof: We begin by following the same method of Theorem 4.2.1. Cases (i) and (ii) will be similar, but the method of Theorem 4.2.1 will fail after that. The c-differential equation for the inverse function with a linearized polynomial is

$$(x+a)^{p^n-2} + L(x+a) - cx^{p^n-2} - cL(x) = b. \tag{4.15}$$

Case (i). Let $a \neq 0$, $x = 0$. Then Equation (4.15) becomes $a^{-1} + L(a) = b$. Therefore, for any c, and $b = a^{-1} + L(a)$, we have a solution of (4.15).

Case (ii). If $x = -a \neq 0$, then Equation (4.15) transforms into $c\left(a^{-1} + L(a)\right) = b$, which gives us one more solution of (4.15), when $aL(a) + 1 = 0$ and $b = 0$ (c is arbitrary).

Case (*iii*). If $0 \neq x \neq -a$, then Equation (4.15) becomes

$$(1-c)x(x+a)L(x) - (b - L(a))x(x+a) + (1-c)x - ca = 0,$$

which has at most $\deg L + 2$ solutions.

Take now $a = 0$, and obtain

$$x\left(xL(x) - \frac{b}{1-c}x + 1\right) = 0. \tag{4.16}$$

Unfortunately, this is the point where the method of Theorem 4.2.1 stops being useful, since we do not have comparable methods to find the number of solutions of an equation involving a more general Dembowski-Ostrom polynomial. However, using the results of [41] highlighted before the theorem we can get meaningful results. First, we follow some of the character theory, as done in [42].

Let $f(x_1, \ldots, x_n) = b$ be an equation with $(x_1, \ldots, x_n) \in \mathbb{F}_q^n$ and b fixed. Then, as discussed in [42], the number of solutions to this equation, call this value $\mathcal{N}_b$, is

$$\mathcal{N}_b = \frac{1}{q} \sum_{x_1,\ldots,x_n \in \mathbb{F}_q} \sum_{\alpha \in \mathbb{F}_q} \chi_1\left(\alpha\left(f(x_1, \ldots, x_n) - b\right)\right).$$

Thus, in addition to $x = 0$, the number of solutions $\mathcal{N}_{b;c}$ of Equation (4.16) is given by

$$q\mathcal{N}_{b;c} = \sum_{\alpha \in \mathbb{F}_q} \sum_{x \in \mathbb{F}_q} \chi_1\left(\alpha(xL(x) + \frac{b}{c-1}x + 1)\right).$$

To arrive at our upper bound, we take $b = 0$. The equation above becomes

$$q\mathcal{N}_{b;c} = \sum_{\alpha \in \mathbb{F}_q} \chi_1(\alpha) \sum_{x \in \mathbb{F}_q} \chi_1\left(\alpha x L(x)\right).$$

We now use [41] as mentioned before the statement of our theorem and Equation (4.12). Applying this to our DO polynomial $\alpha x L(x)$ we get our claimed upper bound. That is, ${}_c\delta_G \leq \sqrt{qN}$, $N = \max_{\alpha \in \mathbb{F}_q}\{N_\alpha\}$.

For the lower bound, we assume that $L(x) = \sum_{i=0}^{n-1} a_i x^{p^i}$ and conditions (4.13) hold. It was

also shown in [41, Theorem 1.5] that if n is even, $\delta = \gcd(s_1, \ldots, s_k)$ and $p^\delta + 1 = p + 1$ divides $(p^i + 1)$, then the above Weil sum is real and consequently it is equal to $\mu_\alpha \sqrt{p^{n+\gamma}}$, where $\mu_\alpha = \pm 1$ and γ is a nonnegative integer. Therefore,

$$ {}_c\delta_G \geq \frac{1}{p^n} \sum_{\alpha \in \mathbb{F}_q} \chi_1(\alpha) \mu_\alpha p^{\frac{\gamma_\alpha}{2}}. $$

We can be more precise and describe μ_α. Let $S_\alpha = \sum_{x \in \mathbb{F}_q} \chi_1 \left(\sum_{i=0}^{n-1} a_i x^{p^i+1} \right)$. By [41, Theorem 1.6], for every α, if $\gamma_\alpha = 0$, then $S_\alpha = (-1)^{\frac{m}{\delta}} p^m$, and if $\gamma_\alpha > 0$, then γ_α is even and $S_\alpha = (-1)^{\frac{m}{\delta} + \frac{\gamma_\alpha}{2}} p^{m + \frac{\delta \gamma_\alpha}{2}}$. Thus,

$$ {}_c\delta_G \geq (-1)^{\frac{m}{\delta}} p^{-m} \sum_{\alpha \in \mathbb{F}_q, \mathcal{N}_\alpha = 1} \chi_1(\alpha) + (-1)^{\frac{m}{\delta}} p^{-m} \sum_{\alpha \in \mathbb{F}_q, \mathcal{N}_\alpha > 1} \chi_1(\alpha) (-1)^{\frac{\gamma_\alpha}{2}} p^{\frac{\delta \gamma_\alpha}{2}}. $$

Our theorem is shown. ■

Example 4.3.1 *Similar to Subsection* 4.2.1 *in the case of monomials, we will use* $p = 2$ *and* $n = 8$. *We must add a polynomial of the form* $L(x) = \sum_{i=0}^{7} a_i x^{2^i}$ *to* $F(x) = x^{2^8-2} = x^{254}$. *We note with* $a_i \in \mathbb{F}_{2^8}$, *there are* $(256)^8 = 2^{64} \approx 1.845e19$ *different potential polynomials with which we could EA transform the inverse function. Recall the DU of the inverse function over* $\mathbb{F}_{2^8}$ *is 4. Using SageMath, we computed the maximum cDU of* $x^{254} + L(x)$ *for several hundred variants. A majority returned a max cDU between 7 and 9, with a high of 12 (tripling traditional DU) and low of 6. An example of a polynomial resulting in 12 is* $L(x) = x^{2^4} + x^{2^5} + x^{2^6}$. *That is, if* $G(x) = x^{254} + L(x)$, *then* ${}_c\delta_G = 12$.

While the theoretical results on polynomials do not provide us with as much insight as results on monomials, we again see that EA transformations have the potential for large changes to the cDU of an (n, n, p)-function. This is in contrast with the regular DU that remains invariant under EA transformations, and this observation concludes our look into the c-differential uniformity of several EA transformations of the inverse function over finite fields. Next, we survey the results of multiple research efforts into functions which may prove most resilient against a potential c-differential attack.

4.4 Summary of Results on c-Differential Uniformity

Since the introduction and publication of the c-differential and the corresponding c-differential uniformity in 2020, there have been multiple papers written investigating the cDU properties of different classes of functions, mostly with the goal of finding functions with low cDU. In this section, we compile and summarize findings to date in this emerging area of cryptologic research. A table is provided for quick reference.

In [2], the article introducing the c-differential as a potential expansion of differential cryptanalysis, the authors consider how known perfect nonlinear functions (recall PN functions do not exist for $p = 2$) and the inverse function perform under the cDU concept. Soon after, multiple papers followed in quick succession. Power functions with low cDU of the form x^d for various c, d are investigated in [43], [44] and [45]. Also in [45], the authors investigate several classes of almost perfect nonlinear (differential uniformity of 2) functions and demonstrate that the cDU increases significantly in some cases. The particular case of $c = -1$ (also known as quasi-planar functions) is investigated in [33] and [46], and a detailed look into a modified Gold function is done in [47]. Some construction and existence results on PcN and APcN functions are provided in [27], and different multinomial classes of such functions are found in [48].

In Table 4.1, the maximum c-differential uniformity ($_c\delta_F$) of $F : \mathbb{F}_{p^n} \to \mathbb{F}_{p^n}$ is captured for prime p as indicated. The default is condition is $c \neq 1$, and anything additional is captured in the "conditions" column.

Table 4.1. $_c\delta_F$ of Various Classes of Functions, $c \neq 1$

$F(x)$	$\mathbb{F}_{p^n}$	$_c\delta_F$	Conditions	Ref
x^2	$p > 2$	2 (APcN)	none	[2]
$x^{10} - ux^6 - u^2x^2$	$p = 3$	≥ 2	$u \in \mathbb{F}_{3^n}$	[2]
$x^{(3^k+1)/2}$	$p = 3$	1 (PcN)	$c = -1$, $\frac{2n}{\gcd(k,2n)}$ is odd	[2]
x^{p^n-2}	any p	1 (PcN)	$c = 0$	[2]
x^{2^n-2}	$p = 2$	2 (APcN)	$c \neq 0$, $\mathrm{Tr}(c) = \mathrm{Tr}(1/c) = 1$	[2]
x^{2^n-2}	$p = 2$	3	$c \neq 0$, $\mathrm{Tr}(c) = 0$ or $\mathrm{Tr}(1/c) = 0$	[2]

Continued on next page

Table 4.1 – continued from previous page

$F(x)$	$\mathbb{F}_{p^n}$	${}_c\delta_F$	Conditions	Ref
x^{p^n-2}	$p>2$	2 (APcN)	$c\neq 0, (c^2-4c)\notin[\mathbb{F}_{p^n}]^2, (1-4c)\notin[\mathbb{F}_{p^n}]^2$	[2]
x^{p^n-2}	$p>2$	2 (APcN)	$c=4,4^{-1}$	[2]
x^{p^n-2}	$p>2$	3	$c\neq 0,4,4^{-1}$, $(c^2-4c)\in[\mathbb{F}_{p^n}]^2$ or $(1-4c)\in[\mathbb{F}_{p^n}]^2$	[2]
x^{2^k+1}	$p=2$	$\frac{2^{\gcd(2k,n)}-1}{2^{\gcd(k,n)}-1}$	$c\in\mathbb{F}_{2^{\gcd(n,k)}}$, $\frac{n}{\gcd(n,k)}\geq 3 (n\geq 3)$	[45]
x^{2^k+1}	$p=2$	$2^{\gcd(n,k)}+1$	$c\in\mathbb{F}_{2^n}\setminus\mathbb{F}_{2^{\gcd(n,k)}}$	[45]
x^{p^k+1}	any p	$\gcd(p^k+1,p^n-1)$	$c\in\mathbb{F}_{p^{\gcd(n,k)}}$	[45]
$x^{(p^k+1)/2}$	$p>2$	$p^{\gcd(n,k)}+1$	$c=-1$	[45]
$x^{(p^n+1)/2}$	$p>2$	≤ 4	$c\neq\pm 1$	[45]
$x^{(p^n+1)/2}$	$p>2$	≤ 2	$c\neq\pm 1, \eta\left(\frac{1-c}{1+c}\right)=1$ $p^n\equiv 1 \pmod 4$	[45]
$x^{(2p^n-1)/3}$	any	≤ 3	$p^n\equiv 2 \pmod 3$	[45]
$x^{(p^n+3)/2}$	$p>3$	≤ 3	$c=-1$, $p^n\equiv 3 \pmod 4$	[45]
$x^{(p^n+3)/2}$	$p>3$	≤ 4	$c=-1$, $p^n\equiv 1 \pmod 4$	[45]
$x^{(p^n-3)/2}$	$p>2$	≤ 4	$c=-1$	[45]
$x^{(3^n+3)/2}$	$p=3$	2 (APcN)	$c=-1$, n even	[45]
$x^{(3^n-3)}$	$p=3$	6	$c=-1$, $n=0 \pmod 4$	[45]
$x^{(3^n-3)}$	$p=3$	4	$c=-1$, $n\neq 0 \pmod 4$	[45]
$x^{(3^n-3)}$	$p=3$	2 (APcN)	$c=0$,	[45]
x^d	$p=3$	1 (PcN)	n,k odd, $c=-1$, $\gcd(n,k)=1$, $d\equiv\frac{3^n+1}{4}\cdot(\frac{3^k+1}{4})^{-1} \pmod{3^n-1}$	[43]
x^d	$p=5$	1 (PcN)	n,k odd, $c=-1$, $\gcd(n,k)=1$, $d\equiv\frac{5^n-1}{2}+(\frac{5^k+1}{2})^{-1} \pmod{5^n-1}$	[43]
x^d	$p>2$	≤ 6	d even, $c=-1$, $d(p^k+1)\equiv\frac{p^n+1}{2} \pmod{p^n-1}$, $p^n\equiv 3 \pmod 4$	[43]
x^d	$p>2$	≤ 3	d odd, $c=-1$, $p^n\equiv 3 \pmod 4$, $d(p^k+1)\equiv\frac{p^n+1}{2} \pmod{p^n-1}$	[43]
$x^{\frac{p^n+1}{4}+\frac{p^n-1}{2}}$	$p>2$	≤ 3	$c=-1$, $p^n\equiv 7 \pmod 8$	[43]
$x^{\frac{p^n-1}{2}+p^k+1}$	$p>2$	≤ 3	$c=-1$, $\frac{n}{\gcd(n,k)}$ odd, $p^n\equiv 3 \pmod 4$	[43]

Continued on next page

Table 4.1 – continued from previous page

$F(x)$	$\mathbb{F}_{p^n}$	${}_c\delta_F$	Conditions	Ref
$x^{\frac{p^n-1}{2}+p^k+1}$	$p>2$	≤ 6	$c=-1$, $\frac{n}{\gcd(n,k)}$ odd, $p^n \equiv 1 \pmod 4$	[43]
$x^{\frac{p^l+1}{2}}$	$p>2$	1 (PcN)	$c=-1$, $l=0$ or l even and n odd, or l,n both even together with $t_2 \geq t_1+1$, where $n=2^{t_1}u$ and $l=2^{t_2}$ such that $2 \nmid u,v$	[33]
$x^{\frac{p^l+1}{2}}$	$p>2$	$\frac{p+1}{2}$	$c=-1$, $\gcd(l,2n)=1$, $p \equiv 1 \pmod 4$ or $p \equiv 3 \pmod 8$	[33]
$x^{\frac{5^l+1}{2}}$	$p=5$	3	$c=-1$, $\gcd(l,2n)=1$	[33]
$x^{\frac{3^l+1}{2}}$	$p=3$	2 (APcN)	$c=-1$, $\gcd(l,2n)=1$	[33]
$x^{p^4+(p-2)p^2+(p-1)p+1}$	$p>2$	1 (PcN)	$c=-1$, $n=5$	[33]
$x^{\frac{p^5+1}{p+1}}$	$p>2$	1 (PcN)	$c=-1$, $n=5$	[33]
x^d	$p>2$	1 (PcN)	$c=-1$, $d=(p-1)p^6+p^5+(p-2)p^3+(p-1)p^2+p$, $n=7$	[33]
$x^{\frac{p^7+1}{p+1}}$	$p>2$	1 (PcN)	$c=-1$, $n=7$	[33]
$L(x)(\sum_{i=1}^{l-1} L(x)^{\frac{p^n-1}{l}i}+u)$	any p	≤ 2 (APcN)	$L(x)$ a linearized polynomial, $l\|(p^n-1)$, $u \neq 1,(1-l) \mod p$, $1-\frac{l}{(1-c)(u+l-1)}, 1+\frac{l}{(1-c)(u-1)} \in D_0$	[48]
$(x^{p^k}-x)^{\frac{q-1}{2}+1}+a_1x+a_2x^{p^k}+a_3x^{p^{2k}}$	$p=3$	≤ 2 (APcN)	$c=-1$, $0 \leq i \leq 2$, $a_1,a_2,a_3 \in \mathbb{F}_3$, $a_1+a_2+a_3 \neq 0$	[48]
$x^{\frac{p^n+7}{2}}$	$p=3$	≤ 2 (APcN)	$c=-1$, n odd	[48]
$f(x)(\mathrm{Tr}_1^q(x)+1)+f(x+\gamma)\mathrm{Tr}_1^q(x)$	$p=2$	1 (PcN)	$f(x)$ is PcN, $\gamma \in \mathbb{F}_q^*$	[48]
$L(x)+L(\gamma)\mathrm{Tr}_q^{q^n}(x)^{q-1}$	any p	1 (PcN)	$L(x)$ a linearized polynomial over $\mathbb{F}_q$, $\gamma \in \mathbb{F}_q^*$, $\mathrm{Tr}_q^{q^n}(\gamma)=0$	[48]
$u\phi(x)+g(\mathrm{Tr}_q^{q^n}(x))^q-g(\mathrm{Tr}_q^{q^n}(x))$	any p	1 (PcN)	ϕ an $\mathbb{F}_q$ linear polynomial, $g(x) \in \mathbb{F}_{q^n}[x]$, $u \in \mathbb{F}_q^*$, $\ker(\phi) \cap \ker(\mathrm{Tr}_q^{q^n})=\{0\}$	[48]
$u(x^q-x)+g(\mathrm{Tr}_q^{q^n}(x))$	any p	1 (PcN)	$g(x) \in \mathbb{F}_{q^n}[x]$ a permutation of $\mathbb{F}_q$, $u \in \mathbb{F}_q^*$, $p \nmid n$	[48]

Continued on next page

Table 4.1 – continued from previous page

$F(x)$	$\mathbb{F}_{p^n}$	${}_c\delta_F$	Conditions	Ref
x^d	$p = 3$	≤ 2 (APcN)	$c = -1$, $d = (3^{\frac{n+1}{2}-1})/2$, $n \equiv 1 \pmod 4$	[46]
x^d	$p = 3$	≤ 2 (APcN)	$c = -1, d = (3^{\frac{n+1}{2}-1})/2+(3^n-1)/2$, $n \equiv 3 \pmod 4$	[46]
x^d	$p = 3$	≤ 2 (APcN)	$c = -1$, $d = (3^{n+1}-1)/8$, $n \equiv 1 \pmod 4$	[46]
x^d	$p = 3$	≤ 2 (APcN)	$c = -1, d = (3^{n+1}-1)/8+(3^n-1)/2$, $n \equiv 3 \pmod 4$	[46]
x^d	$p = 3$	≤ 4	$c = -1$, $d = (3^{\frac{n+1}{4}}-1)(3^{\frac{n+1}{2}}+1)$, $n \equiv 3 \pmod 4$	[46]
x^d	$p = 3$	≤ 4	$c = -1, d = (3^n+1)/4+(3^n-1)/2$, n odd	[46]
$x^{d^{-1}}$	any p	1 (Pc'N)	x^d is PcN, $c' = c^d$, d^{-1} is the inverse of $d \pmod{p^n-1}$	[44]
x^d	$p = 2$	1 (PcN)	$d = 2^j$ for $0 \leq j \leq n-1$, or $d \in \{2^j(2^k+1), 0 \leq j \leq n-1\}$, $k \in \mathbb{Z}^+$	[44]
x^d	$p > 2$	1 (PcN)	$c = -1, d(p^k+1) \equiv 2 \pmod{p^n-1}$, $p^n \equiv 3 \pmod 4$, d odd	[44]
x^d	$p > 2$	1 (PcN)	$c = -1, d \cdot \frac{p^k+1}{2} \equiv \frac{p^n+1}{2} \pmod{p^n-1}$, $v_2(k) = v_2(n)$,$p^n \equiv 1 \pmod 4$	[44]

Many of the functions captured in Table 4.1 have low cDU under the specified conditions, as most of the papers were interested in finding functions that are resistant to any potential form differential cryptanalysis based on the c-differential. The exceptions are from [45], where the authors demonstrated how several APN functions see their cDU increase from 2 when $c = 1$ to a much larger number, depending on the conditions. Additionally, as we have shown with the inverse function, certain transformations do not in general preserve the low cDU of a function. A potential area of future research is to consider certain transformations of the functions captured in Table 4.1 and determine their performance.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 5:
Higher Order c-Derivatives and Differentials

The derivatives of Boolean, p-ary, and vectorial functions defined in Chapter 2 are themselves functions which can be differentiated again. This leads to the concept of higher order discrete derivatives and higher order differential cryptanalysis. Inspired by [20], in which higher order derivatives of functions between Abelian groups are proposed and their applications to cryptography are discussed, in this chapter we extend the c-derivative into higher order, derive some of its properties, and investigate second order c-differential uniformity of the inverse function. Material in this chapter is based on Geary, Calderini, Riera, and Stănică [49].

5.1 Definition and Some Properties

We naturally extend the c-derivative to higher order with the following definition:

Definition 5.1.1 *Let $F : \mathbb{F}_{p^n} \to \mathbb{F}_{p^m}$ be an (n, m, p)-function. The i-th c-derivative of F at $(a_1, a_2, \ldots, a_i)$ is*

$$ {}_cD^{(i)}_{a_1,\ldots,a_i}F(x) = {}_cD_{a_i}({}_cD^{(i-1)}_{a_1,\ldots,a_{i-1}}F(x)) $$

where ${}_cD^{(i-1)}_{a_1,\ldots,a_{i-1}}F(x)$ is the $(i-1)$-th derivative of F at $(a_1, a_2, \ldots, a_{i-1})$.

This implies the 0-th c-derivative is the function F itself and the 1st c-derivative is the c-derivative defined in Chapter 3 and used to generalize multiple cryptographic properties of vectorial Boolean and p-ary functions. Notice that when $c = 1$, we recover the traditional (n, m, p)-function higher order derivative.

Before we explore these new higher order derivatives we need to ensure several basic properties carry over from the traditional (i.e., $c = 1$) case. First, we see that the sum rule

holds. That is, that the c-derivative of a sum is a sum of the c-derivatives.

$$\begin{aligned} {}_cD_a(F+G)(x) &= F(x+a) + G(x+a) - c(F(x) + G(x)) \\ &= F(x+a) - cF(x) + G(x+a) - cG(x) \\ &= {}_cD_aF(x) + {}_cD_aG(x). \end{aligned}$$

A product rule exists for the traditional derivative, $D_a(FG)(x) = F(x+a)D_aG(x) + D_aF(x)G(x)$. We find something similar with the c-derivative,

$$\begin{aligned} {}_cD_a(FG)(x) &= F(x+a)G(x+a) - cF(x)G(x) \\ &= F(x+a)\left(G(x+a) - cG(x)\right) + \left(\left(F(x+a) - F(x)\right)cG(x)\right. \\ &= F(x+a)\ {}_cD_aG(x) + \ {}_cD_aF(x)\ c\,G(x). \end{aligned}$$

Now, we consider the higher order c-derivatives. When $i = 2$ we have

$$\begin{aligned} {}_cD^{(2)}_{a_1,a_2}F(x) =& {}_cD_{a_2}({}_cD_{a_1}F(x)) \\ =& {}_cD_{a_2}(F(x+a_1) - cF(x)) \\ =& F(x+a_1+a_2) - cF(x+a_2) - c(F(x+a_1) - cF(x)) \\ =& F(x+a_1+a_2) - cF(x+a_2) - cF(x+a_1) + c^2F(x). \end{aligned}$$

Taking another iteration, we have

$$\begin{aligned} {}_cD^{(3)}_{a_1,a_2,a_3}F(x) = & F(x+a_1+a_2+a_3) \\ & - c\left[F(x+a_1+a_2) + F(x+a_1+a_3) + F(x+a_2+a_3)\right] \\ & + c^2\left[F(x+a_1) + F(x+a_2) + F(x+a_3)\right] \\ & - c^3F(x). \end{aligned}$$

We see a similar pattern to Proposition 1 in [20], albeit with the additional complication of

powers of c, and we find the following identity:

$$\begin{aligned} F(x+a_1+a_2+a_3) = {}& {}_cD^{(3)}_{a_1,a_2,a_3}F(x) \\ & + c\left[{}_cD^{(2)}_{a_1,a_2}F(x) +_c D^{(2)}_{a_1,a_3}F(x) +_c D^{(2)}_{a_2,a_3}F(x)\right] \\ & + c^2\left[{}_cD_{a_1}(F(x)) +_c D_{a_2}(F(x)) +_c D_{a_3}(F(x))\right] \\ & + c^3F(x). \end{aligned}$$

The pattern holds in general, as we now show.

Theorem 5.1.2 *Let F be an (n,m,p)-function with ${}_cD^{(i)}_{a_1,\ldots,a_i}F(x)$ the i-th c-derivative of F at $(a_1,a_2,\ldots,a_i)$. Then*

$$\begin{aligned} F(x+a_1+a_2+\cdots+a_n) = {}& c^nF(x) \\ & + c^{n-1}\left[\sum_{i=1}^{n} {}_cD_{a_i}F(x)\right] \\ & + c^{n-2}\left[\sum_{1\le j_1<j_2\le n} {}_cD^2_{a_{j_1},a_{j_2}}F(x)\right] \\ & + \ldots \\ & + c^2\left[\sum_{1\le j_1<\ldots<j_i\le n} {}_cD^{(n-2)}_{a_{j_1},\ldots,a_{j_i}}F(x)\right] \\ & + c^1\left[\sum_{1\le j_1<\ldots<j_i\le n} {}_cD^{(n-1)}_{a_{j_1},\ldots,a_{j_i}}F(x)\right] \\ & + c^0\left[{}_cD^{(n)}_{a_1,\ldots,a_n}F(x)\right]. \end{aligned}$$

In other words,

$$F(x+a_1+a_2+\cdots+a_n) = \sum_{i=0}^{n}\sum_{1\le j_1<\ldots<j_i\le n} c^{n-i}\ {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i}}F(x). \tag{5.1}$$

Proof: Equation 5.1 can also be written as

$$F(x+a_1+a_2+\cdots+a_n) = {}_cD^{(n)}_{a_1,\ldots,a_n}F(x) + \sum_{i=0}^{n-1}\sum_{1\le j_1<\ldots<j_i\le n-1} c^{n-1-i}\ {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i}}F(x),$$

which implies

$${}_cD^{(n)}_{a_1,\ldots,a_n}F(x) = F(x+a_1+a_2+\cdots+a_n) - \sum_{i=0}^{n-1}\sum_{1\le j_1<\ldots<j_i\le n-1} c^{n-1-i}\ {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i}}F(x).$$

We proceed by induction. For $n = 1$, we see (5.1) follows directly from the definition and $n = 2, 3$ can be seen in the discussion before the theorem. Assuming Equation 5.1 holds for $n-1$, we have

$$\begin{aligned}
{}_cD^{(n)}_{a_1,\ldots,a_n}F(x) &= {}_cD_{a_n}\Big({}_cD^{(n-1)}_{a_1,\ldots,a_{n-1}}F(x)\Big)\\
&= {}_cD_{a_n}\Big(F(x+a_1+\cdots+a_{n-1}) - \sum_{i=0}^{n-2}\sum_{1\le j_1<\ldots<j_i\le n-2} c^{n-2-i}\ {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i}}F(x)\Big)\\
&= F(x+a_1+\cdots+a_n) - cF(x+a_1+\cdots+a_{n-1})\\
&- {}_cD_{a_n}\Big(\sum_{i=0}^{n-2}\sum_{1\le j_1<\ldots<j_i\le n-2} c^{n-2-i}\ {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i}}F(x)\Big).
\end{aligned}$$

We apply the induction hypothesis to $cF(x+a_1+\cdots+a_{n-1})$, and noticing the last double sum is composed of all the c-derivatives that include a_n, we have

$$\begin{aligned}
&F(x+a_1+\cdots+a_n) - cF(x+a_1+\cdots+a_{n-1})\\
&- {}_cD_{a_n}\Big(\sum_{i=0}^{n-2}\sum_{1\le j_1<\ldots<j_i\le n-2} c^{n-2-i}\ {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i}}F(x)\Big)\\
&= F(x+a_1+\cdots+a_n) - c\Big(\sum_{i=0}^{n-2}\sum_{1\le j_1<\ldots<j_i\le n-2} c^{n-2-i}\ {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i}}F(x)\Big)\\
&- \Big(\sum_{i=0}^{n-1}\sum_{1\le j_1<\ldots<j_i\le n-1} c^{n-1-i}\ {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i},a_n}F(x)\Big)
\end{aligned}$$

$$= F(x + a_1 + \cdots + a_n) - \Bigg(\Big(\sum_{i=0}^{n-2} \sum_{1 \le j_1 < \ldots < j_i \le n-2} c^{n-1-i} \, {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i}} F(x) \Big)$$
$$+ \Big(\sum_{i=0}^{n-1} \sum_{1 \le j_1 < \ldots < j_i \le n-1} c^{n-1-i} \, {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i},a_n} F(x) \Big) \Bigg)$$
$$= F(x + a_1 + \cdots + a_n) - \Big(\sum_{i=0}^{n-1} \sum_{1 \le j_1 < \ldots < j_i \le n-1} c^{n-1-i} \, {}_cD^{(i)}_{a_{j_1},\ldots,a_{j_i}} F(x) \Big).$$

The claim is shown. ■

In Corollary 3.3.5 of Chapter 3, we proved permutation polynomials are perfect 0-nonlinear and vice versa. Now, with Theorem 5.1.2, we can improve upon this result, but first we need to extend c-differential uniformity to higher order.

Definition 5.1.3 *Let F be an (n, n, p)-function with $a_1, a_2, \ldots, a_k, b, c \in \mathbb{F}_{p^n}$, and let ${}_c\Delta_F(a_1, a_2, ..., a_k, b) = |\{x \in \mathbb{F}_{p^n} : {}_cD^{(k)}_{a_1,...,a_k} F(x) = b\}|$. The quantity ${}_c\delta^{(k)}_F = \max\{{}_c\Delta_F(a_1, a_2, ..., a_k, b)\}$ is called the k-*th order c-differential uniformity *of F.*

As in the case of first order cDU, the k-th order cDU is counting the maximum number of solutions of a discrete c-differential equation, now in higher order: ${}_cD^{(k)}_{a_1,...,a_k} F(x) = b$. With this, we can improve upon Corollary 3.3.5.

Corollary 5.1.4 *Let F be an (n, n, p)-function. Then F is a k-th order perfect 0-nonlinear function for all $k \ge 0$ if and only if F is a permutation polynomial.*

Proof: From Theorem 5.1.2 we see that the k-th 0-derivative of a function F is $F(x + a_1 + a_2 + \cdots + a_k)$, which is bijective if and only if F is bijective. Thus permutations have bijective k-th c-derivatives for all k when $c = 0$. ■

While we have shown several properties of the c-derivative closely align with the traditional derivative, one key property does not follow. A fundamental property of traditional derivatives is that the degree of a polynomial function is reduced by at least one for every derivative taken. That is, $\deg(D_a F(x)) \le \deg(F(x)) - 1$. This is not always true in the case

of c-derivatives when $c \neq 1$. For example, consider the linearized monomial $F(x) = x^{p^k}$ over $\mathbb{F}_{p^n}$ with k an integer between 0 and n. This function has degree 1 (recall that the p-ary weight of p^k is 1) and the c-derivative of F at a is $(x+a)^{p^k} - cx^{p^k} = (1-c)x^{p^k} + a^{p^k}$, which is also of degree 1 for all $c \neq 1$. Thus, the reduction of degree is not a general property of the c-derivative.

Next, we show that higher order c-derivatives are invariant under permutation of the a_i's.

Proposition 5.1.5 *Let $F : \mathbb{F}_{p^n} \to \mathbb{F}_{p^n}$, denote $[k] := \{1, \ldots, k\}$, and let $|I|$ be the cardinality of the subsets $I \subseteq [k]$. Then,*

$$ {}_cD^{(k)}_{a_1,\ldots,a_t} F(x) = \sum_{I \subseteq [k]} (-c)^{k-|I|} F\left(x + \sum_{i \in I} a_i\right). $$

In particular, for any permutation π of $\{1, \ldots, k\}$ we have

$$ {}_cD^{(k)}_{a_1,\ldots,a_k} F(x) = {}_cD^{(k)}_{a_{\pi(1)},\ldots,a_{\pi(k)}} F(x) \ . $$

Proof: Starting with $k = 2$, the right-hand side is computed as follows:

$I = \emptyset \to (-c)^2 F(x) = c^2 F(x)$.

$I = \{1\} \to (-c)^{2-1} F(x + a_1) = -cF(x + a_1)$.

$I = \{2\} \to (-c)^{2-1} F(x + a_2) = -cF(x + a_2)$.

$I = \{1, 2\} \to (-c)^{2-2} F(x + a_1 + a_2) = F(x + a_1 + a_2)$.

Summing together these terms is exactly ${}_cD^{(2)}_{a_1,a_2} F(x)$, and so we have ${}_cD^{(2)}_{a_1,a_2} F(x) = \sum_{I \subseteq [2]} (-c)^{2-|I|} F(x + \sum_{i \in I} a_i)$.

Proceeding by induction, we get

$$
\begin{aligned}
{}_cD^{(k)}_{a_1,\ldots,a_t}F(x) &= {}_cD^{(k-1)}_{a_1,\ldots,a_{k-1}}F(x+a_k) - c\left({}_cD^{(k-1)}_{a_1,\ldots,a_{k-1}}F\right) \\
&= \sum_{I\subseteq[k-1]} (-c)^{(k-1)-|I|}F\left(x + a_k + \sum_{i\in I} a_i\right) - c\sum_{I\subseteq[k-1]} (-c)^{(k-1)-|I|}F\left(x + \sum_{i\in I} a_i\right) \\
&= \sum_{\substack{I'\subseteq[k]\\ k\in I'}} (-c)^{(k-1)-(|I'|-1)}F\left(x + \sum_{i\in I'} a_i\right) + \sum_{\substack{I'\subseteq[k]\\ k\notin I'}} (-c)^{k-|I'|}F\left(x + \sum_{i\in I'} a_i\right) \\
&= \sum_{I\subseteq[k]} (-c)^{k-|I|}F\left(x + \sum_{i\in I} a_i\right).
\end{aligned}
$$

From this, we can see that permuting the elements a_i does not change the value of the higher order c-derivative. ■

One of the key findings of higher order derivatives of binary functions is that if the i inputs are not linearly independent, then the ith derivative is exactly 0. That is, if $a_1, a_2, \ldots, a_i$ are linearly dependent, then $D^{(i)}_{a_1,\ldots,a_i}F(x) = 0$. This limits the number of pairs that can be attempted in a higher order differential attack to the dimension of the vector space and reduces the combinations of differences that can be traced simultaneously. However, this property, and therefore the limits, do not apply for higher order c-derivatives when $c \neq 1$, which can be seen by considering the definition of the c-derivative. If we let $a = 0$, then

$$
{}_cD_0F(x) = F(x+0) - cF(x) = (1-c)F(x).
$$

Thus, even in the extreme case of zero difference between the input pairs, the c-derivative results in a nonzero function. In higher order c-derivatives this property remains true. For example, the 2nd c-derivative has the form

$$
{}_cD^{(2)}_{a_1,a_2}F(x) = F(x+a_1+a_2) - cF(x+a_2) - cF(x+a_1) + c^2F(x).
$$

Even if $a_1 = a_2$ (and thus linearly dependent), the 2nd c-derivative is not identically zero due to the introduction of the c multiplier. This fact increases the input (or output) differences that can be traced through an encryption scheme and potentially increases the vulnerability of a cipher if a c-differential attack is realized in the future.

The final property we show in this section is regarding the non-decreasing behavior of the t-order cDU.

Proposition 5.1.6 *Let F be an (n, m, p)-function, $t \in \mathbb{Z}_+$ (positive integers), and $c \neq 1$, then the t-order c-differential uniformity of F is greater than or equal to its $(t-1)$-order c-differential uniformity, ${}_c\delta_F^{(t)} \geq {}_c\delta_F^{(t-1)}$.*

Proof: Let ${}_c\delta_F^{(t-1)} = \beta$ for an (n, m, p)-function F and any $c \neq 1$. By taking $a_t = 0$, we have ${}_cD^{(t)}_{a_1,a_2,\ldots,a_{t-1},0}F(x) = (1-c)\,{}_cD^{(t-1)}_{a_1,a_2,\ldots,a_{t-1}}F(x)$. With $c \neq 1$, ${}_c\delta_F^{(t)}$ is equal to the maximum number of solutions to ${}_cD^{(t-1)}_{a_1,a_2,\ldots,a_{t-1}}F(x) = \frac{b}{1-c}$. Because ${}_c\delta_F^{(t-1)} = \beta$, we can always find b such that $\frac{b}{1-c}$ has β solutions. Thus ${}_c\delta_F^{(t)}$ is bounded below by β. ■

5.2 Second Order c-Differential Spectrum of the Inverse Function

Next, we consider an example of a higher order c-derivative and compare it to the traditional higher order derivative (i.e., when $c = 1$). The function we investigate is the multiplicative inverse function over finite fields of characteristic 2, a popular function used in S-boxes that was a primary motivator in Chapter 4. Recall this function can be represented by a monomial $F : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$, $F(x) = x^{2^n-2}$. We know from Chapter 4 that the cDU of F is 1 for $c = 0$; and 2 or 3, based on the value of $c \neq 1$. In this section, we count solutions to the second order c-differential equation ${}_cD^{(2)}_{a_1,a_2}F(x) = b$ with $c, a_1, a_2, b \in \mathbb{F}_{2^n}$.

The traditional ($c = 1$) second order differential spectrum of the inverse function over $\mathbb{F}_{2^n}$ was recently investigated in [50]. It was shown that for $n \geq 3$ the number of solutions to $D_{a_1,a_2}x^{2^n-2} = b$ is in the set $\{0, 4, 8\}$ and that there are multiple a_1, a_2, b that provide eight solutions for $n \geq 6$.

With this understanding of the behavior of the second traditional derivative of the inverse function, we now consider the second c-derivative of the inverse function and compare the two. Starting with ${}_cD^{(2)}_{a_1,a_2}F(x) = b$, we have

$$(x + a_1 + a_2)^{2^n-2} + c(x + a_2)^{2^n-2} + c(x + a_1)^{2^n-2} + c^2x^{2^n-2} = b. \tag{5.2}$$

It was shown in [2] that the inverse function has a bijective first c-derivative when $c = 0$. In Chapter 3, we showed that any permutation will have a bijective (i.e., balanced) first c-derivative when $c = 0$. For the second c-derivative of the inverse function, when $c = 0$, we get $(x + a_1 + a_2)^{2^n-2} = b$. If $b = 0$, $x = a_1 + a_2$ is the only solution. If $b \neq 0$, then $x \neq a_1 + a_2$ and $\frac{1}{x+a_1+a_2} = b$. Thus, $x = \frac{1}{b} + a_1 + a_2$ is the only solution and we see when $c = 0$ the second c-derivative of the inverse function is a bijection, as is in the case of the first c-derivative when $c = 0$. In fact, we know this is true for all higher orders as well from Corollary 5.1.4. For $c \neq 0$, we consider multiple cases.

Case (i). Let $a_1 = a_2$. Recall this leads to a trivial result in traditional derivatives. Equation (5.2) becomes $x^{2^n-2} + c^2x^{2^n-2} = b$, that is, $(1 + c^2)x^{2^n-2} = b$. When $b = 0$, $x = 0$ is the only solution. If $b \neq 0$, then $x \neq 0$ and $\frac{1+c^2}{x} = b$ gives us one solution $x = \frac{1+c^2}{b}$.

Case (ii). $a_1 \neq a_2$, and $x = a_1, a_2, a_1 + a_2$, or 0. Equation (5.2) becomes, respectively,

$$\begin{aligned}
&a_2^{2^n-2} + c(a_1 + a_2)^{2^n-2} + c^2a_1^{2^n-2} = b, \text{ or,}\\
&a_1^{2^n-2} + c(a_1 + a_2)^{2^n-2} + c^2a_2^{2^n-2} = b, \text{ or,}\\
&ca_2^{2^n-2} + ca_1^{2^n-2} + c^2(a_1 + a_2)^{2^n-2} = b, \text{ or,}\\
&(a_1 + a_2)^{2^n-2} + ca_2^{2^n-2} + ca_1^{2^n-2} = b.
\end{aligned}$$

When $c = 1$ all four of these solutions are the same and can be true simultaneously. However, when $c \neq 1$ we cannot combine all four of these solutions. In fact, the most that can be combined are two. Consider the solutions for a_1 and a_2 (the first two above). If we could combine these, then we would have,

$$a_2^{2^n-2} + c(a_1 + a_2)^{2^n-2} + c^2a_1^{2^n-2} = a_1^{2^n-2} + c(a_1 + a_2)^{2^n-2} + c^2a_2^{2^n-2},$$

which simplifies to

$$\begin{aligned}
a_2^{2^n-2} + c^2a_1^{2^n-2} &= a_1^{2^n-2} + c^2a_2^{2^n-2}, \text{ or,}\\
(1 + c^2)a_1^{2^n-2} &= (1 + c^2)a_2^{2^n-2}.
\end{aligned}$$

For $c \neq 1$ (which is an assumption throughout), a_1 must equal a_2 which is not true in this

case. Therefore, the solutions cannot be combined, and we have that no more than three solutions can be true simultaneously.

Now, we consider the possibility of combining $x = 0$ with $x = a_1 + a_2$. This gives us

$$ca_2^{2^n-2} + ca_1^{2^n-2} + c^2(a_1 + a_2)^{2^n-2} = (a_1 + a_2)^{2^n-2} + ca_2^{2^n-2} + ca_1^{2^n-2},$$

which simplifies to $c^2(a_1+a_2)^{2^n-2} = (a_1+a_2)^{2^n-2}$. This is only true when $c = 1$ or $a_1 = a_2$, neither of which are allowed in this case. From this we immediately see that we can combine at most two of the solutions in Case (ii). There are only at most four values of c that allow the combination of two of these solutions. As we have seen, there are only four possible combinations (which in some cases might be equal): $x = 0$ and $x = a_1$, $x = 0$ and $x = a_2$, $x = a_1 + a_2$ and $x = a_1$, and $x = a_1 + a_2$ and $x = a_2$.

Let $x = 0$ and $x = a_1$ be both solutions of the equation above. Then,

$$(a_1 + a_2)^{2^n-2} + ca_2^{2^n-2} + ca_1^{2^n-2} = a_2^{2^n-2} + c(a_1 + a_2)^{2^n-2} + c^2a_1^{2^n-2}.$$

Rearranging terms, we arrive at $(1 + c)(a_1 + a_2)^{2^n-2} + (1 + c)a_2^{2^n-2} + c(1 + c)a_1^{2^n-2} = 0$, which, since $c \neq 1$, simplifies to $(a_1 + a_2)^{2^n-2} + a_2^{2^n-2} + ca_1^{2^n-2} = 0$.

If $a_1 = 0$, then $x = 0$ and $x = a_1$ are the same solution, so we can assume that $a_1 \neq 0$. If $a_2 = 0$, we arrive at the equation $(1 + c)a_1^{2^n-2} = 0$, which only has the forbidden solutions $c = 1$ or $a_1 = 0$. We can then assume that $a_1a_2 \neq 0$. The equation becomes then $c(a_1+a_2)a_2+a_1^2 = 0$, which has a single solution $c_0 = \frac{a_1^2}{(a_1+a_2)a_2}$. It is easy to see that $c_0 = 0$ if and only if $a_1 = 0$. However, it is possible to obtain that $c_0 = 1$ if $a_1^2 + a_2^2 + a_1a_2 = 0$, which is achievable only if n is even and $a_1 = a_2\omega$ or $a_1 = a_2\omega^2$, where $\mathbb{F}_4 = \{0, 1, \omega, \omega^2\}$. As long as $n \geq 5$, we can always choose valid a_1, a_2 to ensure that $c_0 \neq 1$. By symmetry, $x = 0$ and $x = a_2$ give $c_1 = \frac{a_2^2}{(a_1+a_2)a_1}$, with the same conditions as $x = 0$ and $x = a_1$.

Now, $x = a_1 + a_2$ and $x = a_1$ give

$$ca_2^{2^n-2} + ca_1^{2^n-2} + c^2(a_1 + a_2)^{2^n-2} = a_2^{2^n-2} + c(a_1 + a_2)^{2^n-2} + c^2a_1^{2^n-2},$$

which, by rearranging, becomes $(1 + c)a_2^{2^n-2} + c(1 + c)a_1^{2^n-2} + c(1 + c)(a_1 + a_2)^{2^n-2} = 0$,

and, since $c \neq 1$, this can be simplified to $a_2^{2^n-2} + ca_1^{2^n-2} + c(a_1+a_2)^{2^n-2} = 0$. If $a_1 = 0$, then we have the case $x = 0, x = a_2$. If $a_2 = 0$, we do not have two different solutions. We can then assume $a_1 a_2 \neq 0$. Then, the equation is equivalent to $(a_1+a_2)a_1 + ca_2^2 = 0$, which has the solution $c_2 = \frac{a_1(a_1+a_2)}{a_2^2}$. It is easy to see that $c_2 \neq 0$, and that $c \neq 1$ under the same conditions as for $x = 0$ and $x = a_1$. By symmetry, $x = a_1 + a_2$ and $x = a_1$ gives $c_3 = \frac{a_2(a_1+a_2)}{a_1^2}$.

Case (iii). $a_1 \neq a_2, x \neq a_1, a_2, a_1 + a_2$, or 0. Equation (5.2) becomes

$$\frac{1}{x+a_1+a_2} + \frac{c}{x+a_1} + \frac{c}{x+a_2} + \frac{c^2}{x} = b.$$

Multiplying through by $(x+a_1+a_2)(x+a_1)(x+a_2)x$, collecting and rearranging terms, we arrive at

$$\begin{aligned} bx^4 + (1+c^2)x^3 + (a_2 + ca_2 + ba_2^2 + a_1 + ca_1 + ba_1^2 + ba_1a_2)x^2 \\ + (a_1a_2 + ca_2^2 + c^2a_2^2 + ca_1^2 + c^2a_1^2 + c^2a_1a_2 + ba_1a_2^2 + ba_1^2a_2)x \\ + c^2a_1a_2(a_1+a_2) = 0. \quad (5.3) \end{aligned}$$

This quartic polynomial has at most four solutions when $b \neq 0$ and at most three when $b = 0$. Without the four guaranteed solutions from Case (ii), we cannot reach the eight solutions possible when $c = 1$. This means that, as in the case of the first c-derivative of the inverse function, when $c \neq 1$ the differential counts *decreases* from the traditional case. In fact, when combining Cases (ii) and (iii), we see a maximum of six solutions is possible if $c \neq 1$.

We capture the preceding arguments in the following theorem.

Theorem 5.2.1 *Let $F : \mathbb{F}_{p^n} \to \mathbb{F}_{p^n}$ and $F(x) = x^{2^n-2}$. Then, for any $c \in \mathbb{F}_{2^n} \setminus \{1\}$, we have ${}_c\delta_F^{(2)} \leq 6$.*

Some computations to demonstrate our findings are listed in Table 5.1. From here, we see that the maximum is attainable, at least for $n = 8, 9$. We conjecture that it is attainable for all $n \geq 8$.

n	$c = 1$	$c \neq 1$	$c = 0$
4	4	5	1
5	4	5	1
6	8	5	1
7	8	5	1
8	8	6	1
9	8	6	1

Table 5.1. Maximum Number of Solutions to ${}_cD^{(2)}_{a_1,a_2}x^{2^n-2} = b$

5.3 Summary

In this chapter, we investigated higher order c-derivatives and differentials, noting that traditional derivatives are a special case of our extension (i.e., when $c = 1$). We demonstrated some properties of these higher order derivatives and compared them to the original higher order derivatives of (n, m, p)-functions. We also looked at the specific case of the inverse function over fields of even characteristic. While many properties of higher order c-differentials are preserved from the traditional higher order derivative, a key difference arises in that the higher order c-derivatives do not require linearly independent input differences. Thus, the higher order c-derivatives we have introduced could potentially allow the use of more input pairs (for encryption or decryption), which in turn could lead to differentials with higher probabilities than traditional higher order differential attacks or the new c-differential attack using one derivative.

CHAPTER 6:
Analysis of Known Substitution Boxes

Chapter 4 explored the new c-differential uniformity property of vectorial Boolean and p-ary functions and demonstrated how, when we move from the classical case of $c = 1$ to the more general case of $c \in \mathbb{F}_{p^m}$, the differential count can change significantly. In this chapter, we use the SageMath mathematical software system to explore the cDU properties of some real-world cipher substitution boxes in their published form, and also under certain extended affine equivalences. The resulting data, summarized at the end of the chapter, provides insight into how the S-boxes of ciphers currently in use perform against c-differentials. Similar to our investigation into particular EA equivalent inverse functions in Chapter 4, we add linearized monomials of the form x^{p^t} to the univariate polynomial representation of each substitution box (S-box) and document how the cDU changes.

The S-boxes we analyze are binary (n, n)-functions, with sizes of 4, 5, 6, and 8 bits. In addition to computing the maximum cDU of S-boxes in the encryption process, we also compute the maximum cDU of the S-box decryption process, as a differential attack could be conducted with chosen ciphertexts in addition to the usual chosen plaintext method. We demonstrate that most ciphers perform well under c-differential analysis. However, in some cases moving from $c = 1$ to a general c in $\mathbb{F}_{2^n}$ increases the differential uniformity a significant amount, while in other cases it stays the same or even decreases. We highlight the most interesting results after the data and computations have been presented. Throughout the chapter, we use $S(x)$ to represent the forward (encryption) action of an S-box and $S^{-1}(x)$ to represent the reverse (decryption) action.

6.1 4-bit S-boxes

For all 4-bit S-boxes, we use the finite field $\mathbb{F}_{2^4}$ represented by $\mathbb{F}_2[x]/\langle x^4 + x + 1\rangle$ with "a" a root of $x^4 + x + 1$.

6.1.1 PRESENT

PRESENT is a 64-bit block cipher and was adopted as an International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) lightweight cryptography standard in 2012 [51]. The nonlinear component is based on a 4-bit S-box, presented in hexadecimal format as:

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Table 6.1. The S-box of PRESENT

SageMath computations

Univariate polynomials:

$$S(x) = \left(a^3+a^2+1\right)x^{14} + \left(a^3+a^2+1\right)x^{13} + \left(a^3+a^2\right)x^{12} + \left(a^3+a^2+a\right)x^{11} + \left(a^3+1\right)x^{10} + \left(a^3+1\right)x^9 + \left(a^2+a+1\right)x^8 + a^2x^7 + \left(a^3+a^2\right)x^6 + \left(a^3+a\right)x^5 + \left(a^3+a^2+a\right)x^4 + \left(a^2+a+1\right)x^3 + \left(a^2+a+1\right)x^2 + a^3 + a^2$$

$$S^{-1}(x) = \left(a^3+a^2+1\right)x^{14} + (a+1)x^{13} + \left(a^3+a\right)x^{12} + \left(a^3+1\right)x^{11} + \left(a^2+1\right)x^{10} + (a+1)x^9 + \left(a^3+1\right)x^7 + \left(a^3+1\right)x^5 + ax^4 + \left(a^3+a^2+1\right)x^3 + \left(a^3+a\right)x^2 + \left(a^2+1\right)x + a^2 + 1$$

	$S(x)$	$S^{-1}(x)$
DU	4	4
max cDU	4	6
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 3$	5	6

Table 6.2. cDU of PRESENT

6.1.2 RECTANGLE

RECTANGLE is another 64-bit lightweight block cipher [52]. The 4-bit S-box that provides the nonlinear component is presented in hexadecimal format as:

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	6	5	C	A	1	E	7	9	B	0	3	D	8	F	4	2

Table 6.3. The S-box of RECTANGLE

SageMath computations

Univariate polynomials:

$S(x) = \left(a^3 + a^2 + a\right) x^{14} + \left(a^3 + a^2 + 1\right) x^{13} + \left(a^3 + 1\right) x^{12} + x^{11} + \left(a^3 + a^2 + 1\right) x^{10} + a^2 x^9 + \left(a^2 + a\right) x^7 + a^3 x^6 + \left(a^3 + a^2\right) x^5 + a x^4 + a^3 x^3 + \left(a^3 + a^2 + 1\right) x^2 + a^2 x + a^2 + a$

$S^{-1}(x) = \left(a^3 + a^2 + a\right) x^{14} + \left(a^2 + a + 1\right) x^{13} + \left(a^3 + a^2 + a + 1\right) x^{12} + x^{11} + x^{10} + a^3 x^9 + \left(a^2 + a + 1\right) x^8 + \left(a^3 + 1\right) x^7 + \left(a^3 + a\right) x^6 + \left(a^3 + a^2\right) x^5 + \left(a^3 + 1\right) x^4 + \left(a^2 + a\right) x^3 + x^2 + \left(a^2 + 1\right) x + a^3 + 1$

	$S(x)$	$S^{-1}(x)$
DU	4	4
max cDU	5	5
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 3$	7	7

Table 6.4. cDU of RECTANGLE

6.1.3 SERPENT

SERPENT, a 128-bit block cipher, was a finalist in the AES selection contest [53], finishing second only to Rijndael. The nonlinear component contains eight separate 4-bit S-boxes.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S_0(x)$	3	8	F	1	A	6	5	B	E	D	4	2	7	0	9	C
$S_1(x)$	F	C	2	7	9	0	5	A	1	B	E	8	6	D	3	4
$S_2(x)$	8	6	7	9	3	C	A	F	D	1	E	4	0	B	5	2
$S_3(x)$	0	F	B	8	C	9	6	3	D	1	2	4	A	7	5	E
$S_4(x)$	1	F	8	3	C	0	B	6	2	5	4	A	9	E	7	D
$S_5(x)$	F	5	2	B	4	A	9	C	0	3	E	8	D	6	7	1
$S_6(x)$	7	2	C	5	8	4	6	B	E	9	1	F	D	3	A	0
$S_7(x)$	1	D	F	0	E	8	2	B	7	4	C	A	9	3	5	6

Table 6.5. The S-boxes of SERPENT

SageMath computations

Univariate polynomials:

$S_0(x) = \left(a^3+a^2\right)x^{14}+\left(a^3+1\right)x^{13}+\left(a^2+1\right)x^{12}+\left(a^3+1\right)x^{11}+\left(a^3+a^2+1\right)x^{10}+a^3x^8+\left(a^3+a+1\right)x^7+x^6+\left(a^2+a+1\right)x^5+\left(a^3+1\right)x^4+x^3+\left(a^3+a+1\right)x^2+\left(a^3+1\right)x+a+1$

$S_0^{-1}(x) = \left(a^3+a^2\right)x^{14}+\left(a^3+1\right)x^{13}+ax^{12}+\left(a^3+a^2+a\right)x^{11}+(a+1)x^{10}+\left(a^3+a^2+a\right)x^9+a^3x^8+\left(a^3+a+1\right)x^7+\left(a^3+a^2+a\right)x^6+\left(a^2+1\right)x^5+ax^4+\left(a^2+1\right)x^3+(a+1)x^2+\left(a^2+a\right)x+a^3+a^2+1$

$S_1(x) = \left(a^3+a^2\right)x^{13}+\left(a^3+a+1\right)x^{12}+x^{11}+\left(a^2+a+1\right)x^{10}+\left(a^3+a\right)x^9+\left(a^3+a^2+a+1\right)x^8+\left(a^3+a^2+1\right)x^7+\left(a^3+a^2+1\right)x^6+\left(a^3+1\right)x^5+\left(a^3+a^2+a+1\right)x^4+\left(a^2+1\right)x^2+a^2x+a^3+a^2+a+1$

$S_1^{-1}(x) = \left(a^3+a^2+a\right)x^{13}+\left(a^3+a\right)x^{12}+x^{11}+\left(a^3+a^2+a\right)x^{10}+\left(a^3+a^2+a\right)x^9+\left(a^3+a+1\right)x^8+a^3x^7+a^2x^6+\left(a^3+a^2+a\right)x^5+\left(a^3+a^2+a\right)x^4+ax^3+\left(a^3+a+1\right)x^2+\left(a^2+a\right)x+a^2+1$

$S_2(x) = a^3x^{14}+\left(a^2+1\right)x^{12}+\left(a^3+a+1\right)x^{11}+ax^{10}+a^3x^9+\left(a^2+1\right)x^8+\left(a^3+1\right)x^7+a^3x^6+\left(a^3+a^2+1\right)x^5+\left(a^3+a^2+a+1\right)x^4+\left(a^3+a^2+a\right)x^3+\left(a^3+a^2+a+1\right)x^2+\left(a^2+1\right)x+a^3$

$S_2^{-1}(x) = a^3x^{14}+\left(a^3+a^2+1\right)x^{13}+\left(a^3+a^2+1\right)x^{11}+\left(a^2+a\right)x^{10}+\left(a^3+a\right)x^9+ax^8+\left(a^3+a\right)x^6+\left(a^2+a+1\right)x^5+\left(a^2+a\right)x^3+x^2+\left(a^3+1\right)x+a^3+a^2$

$$S_3(x) = \left(a^2+1\right)x^{14} + \left(a^3+a^2+a\right)x^{13} + \left(a^3+a^2\right)x^{12} + \left(a^3+a\right)x^{11} + a^3x^{10} + ax^9 + x^8 + \left(a^3+a^2+1\right)x^7 + \left(a^3+a^2+a\right)x^6 + \left(a^3+a^2\right)x^5 + \left(a^3+1\right)x^4 + \left(a^3+a\right)x^3 + (a+1)x^2 + \left(a^2+a\right)x$$

$$S_3^{-1}(x) = \left(a^2+1\right)x^{14} + \left(a^3+a^2+a\right)x^{13} + x^{12} + \left(a^3+a^2\right)x^{11} + \left(a^2+a+1\right)x^9 + \left(a^3+a+1\right)x^8 + \left(a^3+a^2+1\right)x^7 + \left(a^2+a\right)x^6 + (a+1)x^5 + \left(a^3+a+1\right)x^4 + ax^3 + \left(a^3+a\right)x^2 + a^3x$$

$$S_4(x) = x^{14} + \left(a^3+a^2\right)x^{13} + \left(a^2+a\right)x^{12} + \left(a^2+a\right)x^{10} + ax^9 + (a+1)x^8 + \left(a^3+1\right)x^7 + \left(a^3+a^2+a+1\right)x^6 + \left(a^3+a\right)x^5 + a^3x^3 + \left(a^3+1\right)x^2 + \left(a^3+a^2+a+1\right)x + 1$$

$$S_4^{-1}(x) = x^{14} + \left(a^3+a^2+1\right)x^{13} + \left(a^2+1\right)x^{12} + \left(a^3+a\right)x^{10} + \left(a^3+a\right)x^9 + \left(a^3+a^2\right)x^8 + a^3x^7 + \left(a^3+a^2+1\right)x^6 + (a+1)x^5 + \left(a^3+a^2+a\right)x^3 + (a+1)x^2 + \left(a^3+a+1\right)x + a^2 + 1$$

$$S_5(x) = a^3x^{14} + x^{13} + \left(a^3+a\right)x^{12} + \left(a^3+a^2+a+1\right)x^{11} + x^{10} + \left(a^3+a^2+a\right)x^9 + \left(a^3+a^2+a\right)x^7 + \left(a^3+a+1\right)x^6 + \left(a^3+1\right)x^5 + \left(a^3+1\right)x^4 + \left(a^3+a\right)x^2 + \left(a^2+a\right)x + a^3 + a^2 + a + 1$$

$$S_5^{-1}(x) = a^3x^{14} + \left(a^3+a+1\right)x^{13} + \left(a^3+a\right)x^{12} + a^3x^{11} + ax^9 + a^3x^8 + x^7 + a^3x^6 + \left(a^3+a^2+a\right)x^5 + a^2x^4 + a^2x^3 + \left(a^3+a^2+1\right)x^2 + \left(a^2+a\right)x + a^3$$

$$S_6(x) = a^3x^{14} + x^{13} + a^3x^{12} + a^3x^{11} + a^3x^{10} + \left(a^2+a\right)x^9 + (a+1)x^8 + \left(a^3+a^2\right)x^7 + \left(a^3+a^2+1\right)x^6 + \left(a^3+a^2+a+1\right)x^5 + \left(a^3+a^2\right)x^4 + \left(a^3+a+1\right)x^3 + \left(a^3+a^2+1\right)x^2 + \left(a^2+1\right)x + a^2 + a + 1$$

$$S_6^{-1}(x) = a^3{*}x^14 + (a^3{+}a^2{+}a{+}1){*}x^13 + (a{+}1){*}x^12 + (a^3{+}a^2{+}a{+}1){*}x^11 + (a^3{+}a^2{+}a{+}1){*}x^10 + (a^3{+}a^2{+}a){*}x^8 + x^7 + a^3{*}x^6 + (a^3{+}1){*}x^5 + (a^3{+}a){*}x^4 + a^3{*}x^3 + (a^3{+}a^2{+}1){*}x^2 + a^3 + a^2 + a + 1$$

$$S_7(x) = ax^{14} + \left(a^3+a^2\right)x^{13} + a^2x^{12} + \left(a^2+1\right)x^{11} + \left(a^3+1\right)x^{10} + \left(a^3+a+1\right)x^9 + \left(a^3+a^2\right)x^8 + \left(a^2+a+1\right)x^7 + a^3x^6 + x^5 + \left(a^3+a\right)x^4 + ax^3 + \left(a^3+a^2+a+1\right)x^2 + a^2x + 1$$

$$S_7^{-1}(x) = ax^{14} + \left(a^2+a\right)x^{13} + a^2x^{11} + \left(a^3+a+1\right)x^{10} + ax^9 + \left(a^3+a\right)x^8 + a^3x^7 + \left(a^3+a^2+a+1\right)x^6 + \left(a^3+a^2+a\right)x^5 + \left(a^3+1\right)x^3 + \left(a^3+a+1\right)x^2 + \left(a^3+a+1\right)x + a + 1$$

	$S_0(x)$	$S_0^{-1}(x)$	$S_1(x)$	$S_1^{-1}(x)$	$S_2(x)$	$S_2^{-1}(x)$	$S_3(x)$	$S_3^{-1}(x)$
DU	4	4	4	4	4	4	4	4
max cDU	5	4	4	5	5	5	6	5
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 3$	6	6	6	6	5	5	5	6

Table 6.6. cDU of SERPENT S-boxes $S_0 - S_3$

	$S_4(x)$	$S_4^{-1}(x)$	$S_5(x)$	$S_5^{-1}(x)$	$S_6(x)$	$S_6^{-1}(x)$	$S_7(x)$	$S_7^{-1}(x)$
DU	4	4	4	4	4	4	4	4
max cDU	5	4	5	5	5	5	5	4
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 3$	6	5	5	5	5	5	5	6

Table 6.7. cDU of SERPENT S-boxes $S_4 - S_7$

6.1.4 SC2000 4

SC2000 is a 128-bit block cipher that utilizes 3 S-boxes: one 4-bit, one 5-bit and one 6-bit [54]. The 4-bit S-box has the following hexadecimal format:

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	2	5	A	C	7	F	1	B	D	6	0	9	4	8	3	E

Table 6.8. The S-box of SC2000 4

SageMath computations

Univariate polynomials:

$S(x) = \left(a^3+1\right)x^{12} + ax^{11} + \left(a^3+a^2+a\right)x^{10} + \left(a^3+a+1\right)x^9 + \left(a^3+a^2+a+1\right)x^8 + x^7 + \left(a^3+1\right)x^6 + \left(a^3+a^2+a\right)x^5 + \left(a^3+a^2\right)x^4 + \left(a^2+a\right)x^3 + a^2x^2 + \left(a^3+a^2+a\right)x + a$

$S^{-1}(x) = x^{13} + (a+1)x^{11} + ax^{10} + a^2x^9 + \left(a^3+a^2\right)x^8 + a^3x^6 + \left(a^2+a+1\right)x^5 + \left(a^2+a\right)x^4 + \left(a^3+a^2+1\right)x^3 + ax^2 + ax + a^3 + a$

	$S(x)$	$S^{-1}(x)$
DU	4	4
max cDU	5	5
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 3$	6	5

Table 6.9. cDU of SC2000 4

6.1.5 PRINCE

PRINCE is a block cipher that is optimized with respect to latency when implemented in hardware [55]. The 4-bit S-box in hexadecimal format is:

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	B	F	3	2	A	C	9	1	6	7	8	0	E	5	D	4

Table 6.10. The S-box of PRINCE

SageMath computations

Univariate polynomials:

$S(x) = \left(a^2+a+1\right)x^{14} + \left(a^3+a\right)x^{13} + a^3x^{12} + \left(a^2+a+1\right)x^{11} + \left(a^3+a^2+a\right)x^{10} + a^3x^9 + \left(a^2+a+1\right)x^8 + x^7 + \left(a^2+a+1\right)x^6 + \left(a^3+a^2+a\right)x^5 + \left(a^2+a+1\right)x^4 + \left(a^3+a+1\right)x^3 + ax^2 + x + a^3 + a + 1$

$S^{-1}(x) = \left(a^2+a+1\right)x^{14} + x^{13} + a^3x^{12} + \left(a^2+a+1\right)x^{11} + \left(a^3+a\right)x^{10} + \left(a^2+a\right)x^9 + \left(a^2+a+1\right)x^8 + \left(a^3+a^2+a+1\right)x^7 + \left(a^3+1\right)x^6 + \left(a^3+a^2\right)x^5 + x^4 + \left(a^3+a\right)x^3 + (a+1)x^2 + \left(a^3+a^2\right)x + a^3 + a + 1$

	$S(x)$	$S^{-1}(x)$
DU	4	4
max cDU	5	5
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 3$	5	7

Table 6.11. cDU of PRINCE

6.2 5-bit S-boxes

For all 5-bit S-boxes, we use the finite field $\mathbb{F}_{2^5}$ represented by $\mathbb{F}_2[x]/\langle x^5+x^2+1\rangle$ with "a" a root of x^5+x^2+1.

6.2.1 FIDES 5

FIDES is lightweight authenticated cipher optimized for hardware implementations [56]. It uses a 5-bit S-box in the 80-bit version and a 6-bit S-box in the 96-bit version. The 5-bit S-box in integer format is:

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(x)$	1	0	25	26	17	29	21	27	20	5	4	23	14	18	2	28
x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$S(x)$	15	8	6	3	13	7	24	16	30	9	31	10	22	12	11	19

Table 6.12. The S-box of FIDES 5

SageMath computations

Univariate polynomials:

$$S(x) = \left(a^4+a^2+a+1\right)x^{24} + \left(a^3+a+1\right)x^{20} + \left(a^4+a\right)x^{18} + (a+1)x^{17} + \left(a^4+a^3+a+1\right)x^{16} + \left(a^3+a\right)x^{12} + \left(a^4+a^3+a\right)x^{10} + \left(a^4+a+1\right)x^9 + \left(a^3+a\right)x^5 + \left(a^4+a^2+a+1\right)x^4 + \left(a^4+a^3+a\right)x^3 + \left(a^4+a^3+a^2+a\right)x^2 + \left(a^3+a^2+1\right)x + 1$$

$$S^{-1}(x) = \left(a^2+a\right)x^{26} + x^{22} + \left(a^4+a+1\right)x^{21} + \left(a^4+1\right)x^{13} + a^2x^{11} + 1$$

	$S(x)$	$S^{-1}(x)$
DU	2	2
max cDU	3	6
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 4$	7	7

Table 6.13. cDU of FIDES 5

6.2.2 DryGASCON

DryGASCON is a 128-bit or 256-bit authenticated encryption with associated data (AEAD) and hashing algorithm [57]. The 128-bit version includes a 5-bit S-box, presented in integer format as:

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(x)$	4	15	27	1	11	0	23	13	31	28	2	16	18	17	12	30
x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$S(x)$	26	25	20	6	21	22	24	10	5	14	9	19	8	3	7	29

Table 6.14. The S-box of DryGASCON

SageMath computations

Univariate polynomials:

$$S(x) = \left(a^4+a^3+a^2\right)x^{24} + \left(a^3+a+1\right)x^{20} + \left(a^4+a^3+a\right)x^{18} + a^3x^{17} + \left(a^2+a\right)x^{16} + \left(a^4+a^3+1\right)x^{12} + \left(a^3+a^2\right)x^{10} + ax^9 + \left(a^4+a^2\right)x^8 + \left(a^4+a^2+a+1\right)x^6 + \left(a^3+a^2+a+1\right)x^4 + \left(a^3+1\right)x^3 + a^3x^2 + \left(a^4+a\right)x + a^2$$

$$S^{-1}(x) = \left(a^4+a^2+a+1\right)x^{28} + \left(a^3+1\right)x^{26} + \left(a^4+a^2\right)x^{25} + \left(a^2+1\right)x^{24} + \left(a^4+a^3+a+1\right)x^{22} + \left(a^2+a\right)x^{21} + \left(a^4+a^2+a\right)x^{20} + \left(a^4+a^3+a\right)x^{19} + (a+1)x^{18} + \left(a^4+a^2\right)x^{17} + ax^{16} + ax^{14} + \left(a^3+a^2+1\right)x^{13} + \left(a^3+a+1\right)x^{12} + \left(a^4+a^2+a\right)x^{11} + \left(a^4+a^3+a+1\right)x^{10} + a^4x^9 + ax^8 + \left(a^4+a^2\right)x^6 + \left(a^4+a^2\right)x^5 + \left(a^3+a\right)x^3 + a^3x^2 + \left(a^4+a^2\right)x + a^2 + 1$$

	$S(x)$	$S^{-1}(x)$
DU	8	8
max cDU	5	6
max cDU with linearized monomial x^{2^t}, $0 \le t \le 4$	7	6

Table 6.15. cDU of DryGASCON

6.2.3 SC2000 5

The 4-bit S-box of SC2000 was analyzed in the previous section. We now consider the 5-bit S-box of SC2000, presented in integer format as:

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(x)$	20	26	7	31	19	12	10	15	22	30	13	14	4	24	9	18
x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$S(x)$	27	11	1	21	6	16	2	28	23	5	8	3	0	17	29	25

Table 6.16. The S-box of SC2000 5

SageMath computations

Univariate polynomials:

$S(x) = \left(a^4+a^2+a\right)x^{28}+\left(a^4+a+1\right)x^{25}+\left(a^4+a^2+a\right)x^{19}+\left(a^4+a^3+a^2+1\right)x^{14}+a^4+a^2$

$S^{-1}(x) = a^3x^{24}+a^3x^{20}+\left(a^4+a+1\right)x^{18}+\left(a^3+a^2+a+1\right)x^{17}+\left(a^3+a^2+1\right)x^{16}+x^{12}+\left(a^4+a^3+a^2+a+1\right)x^{10}+\left(a^3+a^2+1\right)x^9+\left(a^3+a^2\right)x^8+ax^6+x^5+(a+1)x^4+\left(a^2+1\right)x^3+\left(a^3+a^2+1\right)x^2+a^3x+a^4+a^3+a^2$

	$S(x)$	$S^{-1}(x)$
DU	2	2
max cDU	6	3
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 4$	6	5

Table 6.17. cDU of SC2000 5

6.2.4 Shamash

Shamash is a 128-bit authenticated cipher [58] with a 5-bit S-box in integer format as:

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(x)$	16	14	13	2	11	17	21	30	7	24	18	28	26	1	12	6
x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$S(x)$	31	25	0	23	20	22	8	27	4	3	19	5	9	10	29	15

Table 6.18. The S-box of Shamash

SageMath computations

Univariate polynomials:

$S(x) = \left(a^3+a^2+a\right)x^{24}+\left(a^2+1\right)x^{20}+\left(a^3+a^2+a+1\right)x^{18}+\left(a^3+1\right)x^{17}+\left(a^4+1\right)x^{16}+\left(a^2+a\right)x^{12}+\left(a^4+a^3+a+1\right)x^{10}+\left(a^2+a\right)x^9+\left(a^3+a^2\right)x^8+\left(a^3+a\right)x^5+\left(a^4+a^2+1\right)x^4+\left(a^4+1\right)x^3+\left(a^4+a^2+1\right)x^2+\left(a^3+a^2+a\right)x+a^4$

$$S^{-1}(x) = \left(a^4+a^2+1\right)x^{28} + \left(a^4+a^2+1\right)x^{26} + \left(a^3+a^2+1\right)x^{25} + (a+1)x^{24} + \left(a^4+a^3+a^2+1\right)x^{22} + a^3x^{21} + \left(a^4+a^3+a^2+a+1\right)x^{20} + \left(a^4+a+1\right)x^{19} + \left(a^4+a^3+a^2+1\right)x^{18} + ax^{17} + \left(a^2+a+1\right)x^{16} + a^3x^{14} + \left(a^4+a^2\right)x^{13} + \left(a^4+a^3+a^2\right)x^{12} + \left(a^4+a^2+1\right)x^{11} + \left(a^4+a^3\right)x^{10} + \left(a^4+a+1\right)x^9 + x^8 + \left(a^4+a^2+a\right)x^7 + \left(a^3+a^2+a+1\right)x^6 + \left(a^4+a\right)x^5 + x^4 + \left(a^4+a^2\right)x^3 + a^3x^2 + \left(a^4+a^3+1\right)x + a^4 + a$$

	$S(x)$	$S^{-1}(x)$
DU	2	2
max cDU	6	7
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 4$	6	6

Table 6.19. cDU of Shamash

6.3 6-bit S-boxes

For all 6-bit S-boxes, we use the finite field $\mathbb{F}_{2^6}$ represented by $\mathbb{F}_2[x]/\langle x^6+x^4+x^3+x+1\rangle$ with "a" a root of $x^6+x^4+x^3+x+1$.

6.3.1 FIDES 6

The 5-bit S-box of the 80-bit version FIDES of was analyzed in the previous section. Next we consider the 6-bit S-box of the 96-bit version of FIDES, presented in integer format as:

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(x)$	54	0	48	13	15	18	35	53	63	25	45	52	3	20	33	41
x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$S(x)$	8	10	57	37	59	36	34	2	26	50	58	24	60	19	14	42
x	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$S(x)$	46	61	5	49	31	11	28	4	12	30	55	22	9	6	32	23
x	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$S(x)$	27	39	21	17	16	29	62	1	40	47	51	56	7	43	38	44

Table 6.20. The S-box of FIDES 6

SageMath computations

Univariate polynomials:

$$S(x) = \left(a^3+a+1\right)x^{60} + \left(a^5+a^4+a^3\right)x^{58} + a^5x^{57} + \left(a^5+a^3+a^2\right)x^{56} + x^{54} + \left(a^5+a^2+1\right)x^{53} + \left(a^4+a^2+a+1\right)x^{52} + \left(a^5+a^4+a^2\right)x^{51} + \left(a^5+a^4+a^2+a\right)x^{50} + \left(a^5+a^4+a^3+a^2+1\right)x^{49} + \left(a^5+a^4+a^3+a+1\right)x^{48} + \left(a^5+a^4+a^2+a+1\right)x^{46} + \left(a^5+a^3+a+1\right)x^{45} + \left(a^4+a^3+a^2+a+1\right)x^{44} + \left(a^5+a^4+a^3+a^2+a\right)x^{43} + \left(a^4+a^3+a^2+1\right)x^{42} + \left(a^4+a^3+a^2+a\right)x^{41} + \left(a^2+a+1\right)x^{40} + \left(a^3+a^2+1\right)x^{39} + \left(a^3+a^2\right)x^{38} + \left(a^4+a^3\right)x^{37} + (a+1)x^{36} + \left(a^5+a\right)x^{35} + a^4x^{34} + \left(a^5+a^4\right)x^{33} + \left(a^5+a^3+a^2+a+1\right)x^{32} + \left(a^2+1\right)x^{30} + \left(a^4+a^3+a^2+a\right)x^{29} + \left(a^5+a^4+a^3+a^2+1\right)x^{28} + \left(a^3+a+1\right)x^{27} + \left(a^5+a^3\right)x^{26} + \left(a^5+a^3+a\right)x^{25} + \left(a^3+a\right)x^{24} + \left(a^5+a^3+a\right)x^{23} + \left(a^5+1\right)x^{22} + \left(a^3+a+1\right)x^{21} + \left(a^4+a^3\right)x^{20} + \left(a^5+a^3+a\right)x^{19} + \left(a^5+a^4+a^2\right)x^{18} + a^5x^{17} + \left(a^5+a^4+a^3+1\right)x^{16} + \left(a^5+a^4+a^3+a^2\right)x^{15} + \left(a^4+1\right)x^{14} + \left(a^5+a^4+a^3+1\right)x^{13} + \left(a^5+a^4+a^2+1\right)x^{12} + \left(a^3+a\right)x^{11} + \left(a^4+a^3+1\right)x^{10} + \left(a^5+a^3+a+1\right)x^9 + \left(a^5+a^4+a^2+a+1\right)x^8 + \left(a^5+a^2+a\right)x^7 + \left(a^4+a^2\right)x^5 + \left(a^5+a^3+a^2\right)x^4 + \left(a^3+a\right)x^3 + \left(a^5+1\right)x^2 + \left(a^4+1\right)x + a^5+a^4+a^2+a$$

$$S^{-1}(x) = \left(a^5+a^4+a^3+a^2+a+1\right)x^{60} + \left(a^4+a^3+a^2+1\right)x^{58} + \left(a^4+a\right)x^{57} + \left(a^2+a+1\right)x^{56} + \left(a^4+a^3+a^2+1\right)x^{54} + \left(a^5+a^2+1\right)x^{53} + \left(a^5+a^4+a^2\right)x^{52} + \left(a^3+a^2+a\right)x^{51} + \left(a^4+a^3+a+1\right)x^{50} + \left(a^5+a^3+a+1\right)x^{49} + \left(a^5+a^4+a+1\right)x^{48} + \left(a^4+a^3+a^2+a+1\right)x^{46} + \left(a^3+a+1\right)x^{45} + \left(a^2+1\right)x^{44} + \left(a^4+a^3+a^2+a+1\right)x^{43} +$$

$(a^3+a)\,x^{42} + (a^5+a^4+a^2)\,x^{41} + a^2x^{40} + (a^5+a^4+a^2+a)\,x^{39} + (a^4+a^3+1)\,x^{38} + (a^3+1)\,x^{37} + (a^4+a^2+a+1)\,x^{36} + (a^5+a^4+a^2)\,x^{35} + (a^5+1)\,x^{34} + (a^5+a^2+a+1)\,x^{33} + a^4x^{32} + (a^5+a^4+a^3+a^2)\,x^{30} + a^3x^{29} + (a^5+a^4+a^3+1)\,x^{28} + (a^4+a^3+a^2+a)\,x^{27} + (a^4+a^3+a^2+a+1)\,x^{26} + (a^5+a^4+a^3+a+1)\,x^{25} + (a^3+a^2)\,x^{24} + (a^5+a^4)\,x^{23} + (a^5+a^4+a^3)\,x^{22} + (a^4+a^2+1)\,x^{21} + (a^3+a^2)\,x^{20} + (a^4+a^3+a^2+a)\,x^{19} + (a^5+a^4)\,x^{18} + a^2x^{17} + (a^5+a^4+1)\,x^{16} + (a^5+a^4+a^2)\,x^{15} + (a^4+a^3+a^2+a)\,x^{14} + (a^4+a)\,x^{13} + a^3x^{12} + (a^3+a+1)\,x^{11} + (a^2+a)\,x^{10} + (a^5+1)\,x^9 + (a^5+a^2+a+1)\,x^8 + (a^3+a^2+a)\,x^7 + (a^5+a^4+a^2+1)\,x^6 + (a^4+a^3)\,x^5 + (a^5+a^2)\,x^4 + (a^2+1)\,x^3 + (a^5+a^4+a^3+1)\,x^2 + (a^4+a^2+a)\,x + 1$

	$S(x)$	$S^{-1}(x)$
DU	2	2
max cDU	7	6
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 5$	7	8

Table 6.21. cDU of FIDES 6

6.3.2 SC2000 6

The 4 and 5-bit S-boxes of SC2000 were analyzed in the previous sections. The 6-bit S-box has the following integer format:

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(x)$	47	59	25	42	15	23	28	39	26	38	36	19	60	24	29	56
x	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$S(x)$	37	63	20	61	55	2	30	44	9	10	6	22	53	48	51	11
x	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$S(x)$	62	52	35	18	14	46	0	54	17	40	27	4	31	8	5	12
x	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$S(x)$	3	16	41	34	33	7	45	49	50	58	1	21	43	57	32	13

Table 6.22. The S-box of SC2000 6

SageMath computations

Univariate polynomials: $S(x) = \left(a^2+1\right)x^{62} + \left(a^2+a+1\right)x^{61} + \left(a^5+a\right)x^{60} + \left(a^4+a^3+a^2+1\right)x^{59} + \left(a^5+1\right)x^{57} + a^5x^{56} + \left(a^4+a+1\right)x^{55} + \left(a^4+a^3+a^2+1\right)x^{54} + \left(a^5+a^2+a+1\right)x^{53} + \left(a^4+a+1\right)x^{52} + a^4x^{51} + \left(a^5+a^3\right)x^{50} + \left(a^3+a^2\right)x^{49} + a^3x^{48} + \left(a^5+a^4+a^2\right)x^{47} + \left(a^5+a^3+1\right)x^{46} + \left(a^5+a^3+1\right)x^{45} + \left(a^5+a^4+a^3+a^2+a\right)x^{44} + \left(a^3+a^2+a+1\right)x^{43} + \left(a^5+a\right)x^{42} + \left(a^5+a^3+a^2\right)x^{41} + \left(a^5+a^3+a^2\right)x^{40} + \left(a^3+a^2+a\right)x^{39} + \left(a^4+a^3+a^2+a\right)x^{38} + \left(a^5+a^4+a+1\right)x^{37} + \left(a^5+a^2+1\right)x^{36} + \left(a^5+a^4+a^3+a\right)x^{35} + \left(a^4+a^3+a\right)x^{34} + \left(a^2+1\right)x^{33} + \left(a^5+a+1\right)x^{32} + \left(a^2+a+1\right)x^{31} + \left(a^5+a^4+a+1\right)x^{30} + \left(a^4+a^3+a^2+a+1\right)x^{29} + \left(a^5+a^2+a\right)x^{28} + \left(a^4+a^2+1\right)x^{27} + \left(a^5+1\right)x^{26} + \left(a^3+1\right)x^{25} + \left(a^5+a^2+1\right)x^{24} + \left(a^3+a^2+a\right)x^{23} + \left(a^5+a^4+a^3+a^2+a+1\right)x^{22} + \left(a^5+a^4+a+1\right)x^{21} + \left(a^4+a^3\right)x^{20} + a^3x^{19} + \left(a^4+a^3+1\right)x^{18} + \left(a^4+a^3+a^2\right)x^{17} + \left(a^4+a^3+a^2\right)x^{16} + \left(a^4+a^2+1\right)x^{15} + (a+1)x^{14} + \left(a^5+a^3+a^2\right)x^{13} + \left(a^4+a^2+1\right)x^{12} + \left(a^5+a^3+a^2\right)x^{11} + \left(a^4+a^3+a^2+a+1\right)x^{10} + \left(a^5+a^4+a^2+1\right)x^9 + \left(a^5+a^4+a^3+a^2\right)x^8 + \left(a^4+a+1\right)x^7 + \left(a^4+a\right)x^6 + \left(a^5+a^4+a\right)x^5 + \left(a^4+1\right)x^4 + \left(a^4+a^3+a^2\right)x^3 + \left(a^5+a^3+a^2\right)x^2 + a^4x + a^5 + a^3 + a^2 + a + 1$

$S^{-1}(x) = \left(a^2+1\right)x^{62} + \left(a^4+a^2+1\right)x^{61} + \left(a^5+a^3\right)x^{60} + \left(a^5+a+1\right)x^{59} + \left(a^5+a^4+a^3+1\right)x^{58} + \left(a^5+a^4+a^3+a^2+a\right)x^{57} + \left(a^5+a^2+a\right)x^{56} + a^2x^{55} + \left(a^5+a^3\right)x^{54} + \left(a^4+a^2+1\right)x^{53} + \left(a^5+a^4+a^3+a^2+1\right)x^{52} + \left(a^5+a^3+a^2+a\right)x^{51} + \left(a^4+a^2\right)x^{50} + \left(a^4+a^2+a\right)x^{49} + \left(a^5+a^3+a^2\right)x^{48} + \left(a^4+a^3+a^2+a\right)x^{47} + \left(a^5+a^4+a^3+a^2+a\right)x^{46} + \left(a^5+a^3+a^2+1\right)x^{45} + \left(a^5+a^4+1\right)x^{44} + \left(a^4+a^3\right)x^{43} +$

$(a^5+a^4+a^3+1)\,x^{41} + (a^4+a^2)\,x^{40} + (a^3+a^2+a+1)\,x^{39} + (a^5+a^2)\,x^{38} + (a^5+a^4+a^2+a+1)\,x^{37} + (a^5+a^2)\,x^{36} + (a^5+a)\,x^{35} + (a^4+1)\,x^{34} + (a^5+a^4+1)\,x^{33} + (a^5+a^2+a)\,x^{32} + (a^5+a^4+1)\,x^{31} + (a^5+a^3+a)\,x^{30} + (a^3+a)\,x^{29} + a^2x^{28} + a^2x^{27} + (a^4+a^3+a^2+a+1)\,x^{26} + (a^3+a^2)\,x^{25} + (a^3+a^2+1)\,x^{23} + (a^5+a^4+a^3)\,x^{22} + (a^5+a^4+a^3+a)\,x^{21} + (a^5+a^4+a^2+a+1)\,x^{20} + (a^5+a^4+a+1)\,x^{19} + (a^5+a^4+a+1)\,x^{18} + (a^4+a^3+a^2+1)\,x^{17} + (a^4+a^3+a^2+a+1)\,x^{16} + a^4x^{15} + (a^5+a^4+a^3+a^2+a+1)\,x^{14} + (a^5+a^3+1)\,x^{13} + a^4x^{12} + (a^4+a^3+a^2+a+1)\,x^{11} + (a^5+a^2+a+1)\,x^{10} + (a^2+a)\,x^9 + (a^4+a^2+a+1)\,x^8 + (a^5+a^4+a^3+a^2+1)\,x^7 + (a^3+a^2+a+1)\,x^6 + (a^3+a^2+1)\,x^5 + (a^4+a)\,x^4 + (a^4+a^3+1)\,x^3 + (a^4+a^3+a+1)\,x^2 + (a^4+a^3+a^2+a+1)\,x + a^5 + a^2 + a$

	$S(x)$	$S^{-1}(x)$
DU	4	4
max cDU	6	6
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 5$	7	7

Table 6.23. cDU of SC2000 6

6.4 8-bit S-boxes

For all 8-bit S-boxes, we use the finite field $\mathbb{F}_{2^8}$ represented by $\mathbb{F}_2[x]/\langle x^8+x^4+x^3+x^2+1\rangle$ with "a" a root of $x^8+x^4+x^3+x^2+1$. Due to the length of the univariate polynomials for the 8-bit S-boxes, they are not listed in this section, but a few examples are included in the Appendix.

6.4.1 AES

As described in Chapter 4, AES is a 128-bit U.S. federal government block cipher standard. The nonlinear component is an 8-bit S-box based off the inverse function over $\mathbb{F}_{2^8}$, shown in hexadecimal format in Table 6.24. The column of the table is determined by the right four bits and the row is determined by the left 4 bits.

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Table 6.24. The S-box of AES

SageMath computations

Univariate polynomials are listed in the Appendix.

	$S(x)$	$S^{-1}(x)$
DU	4	4
max cDU	9	8
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 7$	9	9

Table 6.25. cDU of AES

6.4.2 Twofish

Twofish was another 128-bit block cipher finalist in the AES competition [59]. The nonlinear component is based on four 8-bit key dependent S-box (KDSB). The main building blocks

of these KDSBs are two 8-bit S-boxes, called q_0 and q_1, which we analyze for c-differential uniformity.

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	a9	67	b3	e8	4	fd	a3	76	9a	92	80	78	e4	dd	d1	38
10	d	c6	35	98	18	f7	ec	6c	43	75	37	26	fa	13	94	48
20	f2	d0	8b	30	84	54	df	23	19	5b	3d	59	f3	ae	a2	82
30	63	1	83	2e	d9	51	9b	7c	a6	eb	a5	be	16	c	e3	61
40	c0	8c	3a	f5	73	2c	25	b	bb	4e	89	6b	53	6a	b4	f1
50	e1	e6	bd	45	e2	f4	b6	66	cc	95	3	56	d4	1c	1e	d7
60	fb	c3	8e	b5	e9	cf	bf	ba	ea	77	39	af	33	c9	62	71
70	81	79	9	ad	24	cd	f9	d8	e5	c5	b9	4d	44	8	86	e7
80	a1	1d	aa	ed	6	70	b2	d2	41	7b	a0	11	31	c2	27	90
90	20	f6	60	ff	96	5c	b1	ab	9e	9c	52	1b	5f	93	a	ef
a0	91	85	49	ee	2d	4f	8f	3b	47	87	6d	46	d6	3e	69	64
b0	2a	ce	cb	2f	fc	97	5	7a	ac	7f	d5	1a	4b	e	a7	5a
c0	28	14	3f	29	88	3c	4c	2	b8	da	b0	17	55	1f	8a	7d
d0	57	c7	8d	74	b7	c4	9f	72	7e	15	22	12	58	7	99	34
e0	6e	50	de	68	65	bc	db	f8	c8	a8	2b	40	dc	fe	32	a4
f0	ca	10	21	f0	d3	5d	f	0	6f	9d	36	42	4a	5e	c1	e0

Table 6.26. The q_0 S-box of Twofish

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	75	f3	c6	f4	db	7b	fb	c8	4a	d3	e6	6b	45	7d	e8	4b
10	d6	32	d8	fd	37	71	f1	e1	30	f	f8	1b	87	fa	6	3f
20	5e	ba	ae	5b	8a	0	bc	9d	6d	c1	b1	e	80	5d	d2	d5
30	a0	84	7	14	b5	90	2c	a3	b2	73	4c	54	92	74	36	51
40	38	b0	bd	5a	fc	60	62	96	6c	42	f7	10	7c	28	27	8c
50	13	95	9c	c7	24	46	3b	70	ca	e3	85	cb	11	d0	93	b8
60	a6	83	20	ff	9f	77	c3	cc	3	6f	8	bf	40	e7	2b	e2
70	79	c	aa	82	41	3a	ea	b9	e4	9a	a4	97	7e	da	7a	17
80	66	94	a1	1d	3d	f0	de	b3	b	72	a7	1c	ef	d1	53	3e
90	8f	33	26	5f	ec	76	2a	49	81	88	ee	21	c4	1a	eb	d9
a0	c5	39	99	cd	ad	31	8b	1	18	23	dd	1f	4e	2d	f9	48
b0	4f	f2	65	8e	78	5c	58	19	8d	e5	98	57	67	7f	5	64
c0	af	63	b6	fe	f5	b7	3c	a5	ce	e9	68	44	e0	4d	43	69
d0	29	2e	ac	15	59	a8	a	9e	6e	47	df	34	35	6a	cf	dc
e0	22	c9	c0	9b	89	d4	ed	ab	12	a2	d	52	bb	2	2f	a9
f0	d7	61	1e	b4	50	4	f6	c2	16	25	86	56	55	9	be	91

Table 6.27. The q_1 S-box of Twofish

SageMath computations

	$S_0(x)$	$S_0^{-1}(x)$
DU	10	10
max cDU	8	8
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 7$	9	8

Table 6.28. cDU of Twofish q_0

	$S_1(x)$	$S_1^{-1}(x)$
DU	10	10
max cDU	7	8
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 7$	10	8

Table 6.29. cDU of Twofish q_1

6.4.3 CLEFIA

CLEFIA is a 128-bit block cipher developed by the Sony Corporation [60]. Two different types of 8-bit S-boxes are employed. S_0 is based on combinations of 4-bit S-boxes, while S_1, similar to the AES S-box, is an affine transformation of the inverse function over $\mathbb{F}_{2^8}$.

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	57	49	d1	c6	2f	33	74	fb	95	6d	82	ea	0e	b0	a8	1c
10	28	d0	4b	92	5c	ee	85	b1	c4	0a	76	3d	63	f9	17	af
20	bf	a1	19	65	f7	7a	32	20	06	ce	e4	83	9d	5b	4c	d8
30	42	5d	2e	e8	d4	9b	0f	13	3c	89	67	c0	71	aa	b6	f5
40	a4	be	fd	8c	12	00	97	da	78	e1	cf	6b	39	43	55	26
50	30	98	cc	dd	eb	54	b3	8f	4e	16	fa	22	a5	77	09	61
60	d6	2a	53	37	45	c1	6c	ae	ef	70	08	99	8b	1d	f2	b4
70	e9	c7	9f	4a	31	25	fe	7c	d3	a2	bd	56	14	88	60	0b
80	cd	e2	34	50	9e	dc	11	05	2b	b7	a9	48	ff	66	8a	73
90	03	75	86	f1	6a	a7	40	c2	b9	2c	db	1f	58	94	3e	ed
a0	fc	1b	a0	04	b8	8d	e6	59	62	93	35	7e	ca	21	df	47
b0	15	f3	ba	7f	a6	69	c8	4d	87	3b	9c	01	e0	de	24	52
c0	7b	0c	68	1e	80	b2	5a	e7	ad	d5	23	f4	46	3f	91	c9
d0	6e	84	72	bb	0d	18	d9	96	f0	5f	41	ac	27	c5	e3	3a
e0	81	6f	07	a3	79	f6	2d	38	1a	44	5e	b5	d2	ec	cb	90
f0	9a	36	e5	29	c3	4f	ab	64	51	f8	10	d7	bc	02	7d	8e

Table 6.30. The S_0 S-box of CLEFIA

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	6c	da	c3	e9	4e	9d	0a	3d	b8	36	b4	38	13	34	0c	d9
10	bf	74	94	8f	b7	9c	e5	dc	9e	07	49	4f	98	2c	b0	93
20	12	eb	cd	b3	92	e7	41	60	e3	21	27	3b	e6	19	d2	0e
30	91	11	c7	3f	2a	8e	a1	bc	2b	c8	c5	0f	5b	f3	87	8b
40	fb	f5	de	20	c6	a7	84	ce	d8	65	51	c9	a4	ef	43	53
50	25	5d	9b	31	e8	3e	0d	d7	80	ff	69	8a	ba	0b	73	5c
60	6e	54	15	62	f6	35	30	52	a3	16	d3	28	32	fa	aa	5e
70	cf	ea	ed	78	33	58	09	7b	63	c0	c1	46	1e	df	a9	99
80	55	04	c4	86	39	77	82	ec	40	18	90	97	59	dd	83	1f
90	9a	37	06	24	64	7c	a5	56	48	08	85	d0	61	26	ca	6f
a0	7e	6a	b6	71	a0	70	05	d1	45	8c	23	1c	f0	ee	89	ad
b0	7a	4b	c2	2f	db	5a	4d	76	67	17	2d	f4	cb	b1	4a	a8
c0	b5	22	47	3a	d5	10	4c	72	cc	00	f9	e0	fd	e2	fe	ae
d0	f8	5f	ab	f1	1b	42	81	d6	be	44	29	a6	57	b9	af	f2
e0	d4	75	66	bb	68	9f	50	02	01	3c	7f	8d	1a	88	bd	ac
f0	f7	e4	79	96	a2	fc	6d	b2	6b	03	e1	2e	7d	14	95	1d

Table 6.31. The S_1 S-box of CLEFIA

SageMath computations

	$S_0(x)$	$S_0^{-1}(x)$
DU	10	10
max cDU	8	7
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 7$	9	9

Table 6.32. cDU of CLEFIA S_0

	$S_1(x)$	$S_1^{-1}(x)$
DU	4	4
max cDU	9	8
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 7$	9	9

Table 6.33. cDU of CLEFIA S_1

6.4.4 SM4

SM4 (formally named SMS4) is a 128-bit Chinese Government block cipher standard [61]. Similar to AES, the nonlinear component is an 8-bit S-box based off the inverse function over $\mathbb{F}_{2^8}$, in hexadecimal format as:

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	D6	90	E9	FE	CC	E1	3D	B7	16	B6	14	C2	28	FB	2C	05
10	2B	67	9A	76	2A	BE	04	C3	AA	44	13	26	49	86	06	99
20	9C	42	50	F4	91	EF	98	7A	33	54	0B	43	ED	CF	AC	62
30	E4	B3	1C	A9	C9	08	E8	95	80	DF	94	FA	75	8F	3F	A6
40	47	07	A7	FC	F3	73	17	BA	83	59	3C	19	E6	85	4F	A8
50	68	6B	81	B2	71	64	DA	8B	F8	EB	0F	4B	70	56	9D	35
60	1E	24	0E	5E	63	58	D1	A2	25	22	7C	3B	01	21	78	87
70	D4	00	46	57	9F	D3	27	52	4C	36	02	E7	A0	C4	C8	9E
80	EA	BF	8A	D2	40	C7	38	B5	A3	F7	F2	CE	F9	61	15	A1
90	E0	AE	5D	A4	9B	34	1A	55	AD	93	32	30	F5	8C	B1	E3
a0	1D	F6	E2	2E	82	66	CA	60	C0	29	23	AB	0D	53	4E	6F
b0	D5	DB	37	45	DE	FD	8E	2F	03	FF	6A	72	6D	6C	5B	51
c0	8D	1B	AF	92	BB	DD	BC	7F	11	D9	5C	41	1F	10	5A	D8
d0	0A	C1	31	88	A5	CD	7B	BD	2D	74	D0	12	B8	E5	B4	B0
e0	89	69	97	4A	0C	96	77	7E	65	B9	F1	09	C5	6E	C6	84
f0	18	F0	7D	EC	3A	DC	4D	20	79	EE	5F	3E	D7	CB	39	48

Table 6.34. The S-box of SM4

SageMath computations

	$S(x)$	$S^{-1}(x)$
DU	4	4
max cDU	9	8
max cDU with linearized monomial x^{2^t}, $0 \leq t \leq 7$	9	9

Table 6.35. cDU of SM4

6.5 Summary of Results

In this chapter, we evaluated S-boxes from multiple real-world ciphers and documented their cDU performance both as published and with certain extended affine transformations. We now highlight some of the findings.

We analyzed 12 different 4-bit S-boxes from five different ciphers. The largest change from traditional DU to cDU was from 4 to 6, which occurred both in PRESENT's S-box and SERPENT's S_6 S-box. The other 10 4-bit S-boxes analyzed remained the same or saw an increase of only 1. With the addition of linearized monomials, two S-boxes (RECTANGLE and PRINCE) had an increase of cDU to 7.

The 5-bit S-boxes produce more interesting results. FIDES 5, Shamash, and SC2000 all have almost perfect nonlinear (APN) S-boxes. That is, DU = 2 when $c = 1$. In the Shamash S-box we see an increase from 2 to 7 in cDU, more than tripling the traditional DU value. For the FIDES 5 and SC2000 S-boxes, the maximum cDU triples to 6. For SC2000 this happens in the forward direction (encryption), for FIDES this happens in the reverse direction (decryption). Thus, the chances of finding large differential characteristics could potential increase significantly in these three cases. Adding a linearized monomial to FIDES bumps the maximum up to 7, while SC2000 stays at a maximum cDU of 6. DryGASCON128, on the other hand, has a DU of 8, and a maximum cDU of 6. The linearized monomial only increases the cDU to 7. Thus, $c = 1$ results in the highest value and the multiplicative differential *decreases* the chance of a finding a high differential characteristic with the DryGASCON S-box, even with the EA transformations we analyzed.

We encounter both FIDES and SC2000 again the in case of 6-bit S-boxes. The 6-bit S-box of FIDES, like the 5-bit version, is APN. This is noteworthy in that to date there is only one known APN permutation, up to affine equivalence, in any even dimension [62]. Therefore, the FIDES 6-bit S-box must be affine equivalent to the APN permutation presented by Dillon et al in [63]. The maximum cDU of the FIDES 6 S-box is 7 and increases slightly to 8 with a linearized monomial. The 6-bit S-box of SC2000 moves from a DU of 4 to a maximum cDU of 6 and with a linearized monomial to 7.

While none of the 8-bit S-boxes we analyzed had the cDU triple in value from DU as we saw in the 5-bit case, several S-boxes saw more than doubling. The S-boxes of AES, SM4, and CLEFIA's S_1 are all affine equivalent to the inverse function and have DU of 4. The max cDU of all three is 9. If we let F be the inverse function over $\mathbb{F}_{2^8}$ and A be an affine transformation of $\mathbb{F}_{2^8}$, then this implies that the affine transformations of these S-boxes must be of the form $A(F(x))$ as we showed earlier transformations of the form $F(A(x))$ preserve cDU. Although $\frac{9}{256}$ is still a small differential, it could potentially result in a higher

differential characteristic of the cipher as a whole. Perhaps surprisingly, considering the results in Chapter 4, the maximum cDU of these three S-boxes did not increase from 9 when adding a linearized monomial. The Twofish building blocks q_0 and q_1 and CLEFIA's S_0 saw a *decrease* in their differential uniformity from a traditional DU of 10, even when adding linearized monomials.

We conclude this computational analysis by noting that even a relatively high maximum cDU value would only be the starting point for a potential c-differential attack and each cipher would need to be considered holistically to determine any future vulnerabilities.

CHAPTER 7:
Conclusion and Future Research

7.1 Conclusion

The goal of this research has been to further the theoretical and practical understanding of the newly proposed c-differential, which has the potential to expand differential cryptanalysis against block ciphers. Using the c-derivative, we modified the autocorrelation function and generalized multiple cryptographic properties of vectorial Boolean and p-ary functions to account for the new differential. This led to the new notions of perfect c-nonlinearity, c-differential bentness, and c-avalanche characteristics, which we investigated in Chapter 3. We showed that our balanced derivative, 0-valued autocorrelation definition of perfect c-nonlinearity is equivalent to our Walsh-Hadamard transform product definition of c-differential bentness, maintaining the traditional equivalence between perfect nonlinearity and bent functions. Our extension of avalanche characteristics includes the notion of c-global avalanche characteristics, where we capture bounds on this new property and expand the idea of how diffusion looks in a substitution box.

Returning to the primary reason the c-differential was introduced in the first place, we explored the c-differential uniformity of the multiplicative inverse function, a popular substitution box primitive, in Chapter 4. We found certain extended affine transformations that, unlike in the case of traditional differential uniformity, significantly increase the maximum value in the cDU spectrum. Thus, if a c-differential attack is ever realized, a cryptographic designer will not be able to substitute certain EA equivalent functions without increasing the vulnerability of an exploitation. The specific EA transformations that resulted in the largest increases were in the form of a linearized monomials, but we also showed there are more general linearized polynomials that can cause large increases in the maximum cDU value.

We then extended c-derivatives and differentials into higher order in Chapter 5, demonstrating several properties and discussing where they diverge from traditional higher derivatives. We returned to the multiplicative inverse function to conduct an analysis of its second order

c-differential spectrum and showed that it is lower valued than the traditional second order differential count.

Finally, we used the SageMath mathematical software system to compute the maximum cDU of many real-world cipher substitution boxes both as published and with the addition of the linearized monomials in Chapter 6. We end with the observation that, if a c-differential attack comes to fruition, the differential characteristics and susceptibility of a cipher to attack will have to be reevaluated using analysis and computations similar to our work in this dissertation.

7.2 Future Research

We see the following lines as potential continuations of this research:

- The cDU of the inverse function significantly increased under certain EA transformations. How do the cDU spectrums of other functions behave under EA transformations?
- The real-world substitution boxes we analyzed for c-differential uniformity were all n by n. However, some popular S-boxes use Feistel network designs that employ n by m S-boxes. How do the differential counts of these functions change with a c multiplier?
- The work in this dissertation focuses on one S-box at a time. How does tracing the resulting cDU through multiple rounds and multiple S-boxes affect the differential characteristics of some recognizable ciphers? In other words, will the linear (diffusion) layers of a cipher allow for a c other than ± 1?
- We analyzed the second order cDU of the inverse function, but how do third and higher orders perform? How do other functions perform with higher order c-differentials?
- We have investigated what happens when we take the c-derivative of a function, but can we go the other way? That is, is there a technique to "integrate" with respect to the c-derivative?

APPENDIX: Univariate Polynomials and SageMath Code

This appendix has two sections. The first section includes a few of the univariate polynomials of 8-bit S-boxes from Chapter 6 that were not displayed due to their length. These examples are provided to demonstrate how complex S-box polynomials can be over finite fields. For this reason and the fact that all values of $c \in \mathbb{F}_{p^n}$ must be evaluated to compute maximum cDU, computations for 8-bit S-boxes proved very time consuming. In the second section, we provide most of the code we used for computations of examples and our analysis of S-boxes.

A.1 8-bit Polynomials: AES

AES is one of the 8-bit S-boxes analyzed in Chapter 6. Recall in SageMath we compute over the finite field $\mathbb{F}_{2^8}$ represented by $\mathbb{F}_2[x]/\langle x^8 + x^4 + x^3 + x^2 + 1\rangle$ with "a" a root of $x^8 + x^4 + x^3 + x^2 + 1$. Both forward (encryption) and reverse (decryption) univariate polynomials are provided below.

$$
\begin{aligned}
S(x) = {} & (a+1)\,x^{254} + \left(a^6 + a^5 + a^2\right) x^{253} + \left(a^7 + a^6 + a^5 + a\right) x^{252} + \left(a^7 + a^5 + a^3 + a + 1\right) x^{251} \\
& + \left(a^7 + a^6 + a^5 + a^3 + a\right) x^{250} + \left(a^7 + a^2 + 1\right) x^{249} + \left(a^4 + a^3 + a^2 + 1\right) x^{248} \\
& + \left(a^7 + a^6 + a^5 + a\right) x^{247} + \left(a^4 + 1\right) x^{246} + \left(a^7 + a^6 + a^5 + a^4 + a^3\right) x^{245} + \left(a^5 + 1\right) x^{244} + \\
& \left(a^7 + a^6 + a^5 + a^4 + a + 1\right) x^{243} + \left(a^7 + a^5 + a^3 + a^2 + a + 1\right) x^{242} + \left(a^7 + a^6 + a^4\right) x^{241} + \\
& \left(a^6 + a^5 + a^4 + a^3 + a^2 + a\right) x^{240} + \left(a^6 + a^4 + a^3 + 1\right) x^{239} + \left(a^7 + a^6 + a^4 + a^3 + a^2 + a\right) x^{238} \\
& + \left(a^7 + a^6 + a^3 + a\right) x^{237} + \left(a^7 + a^6 + a^4 + a + 1\right) x^{236} + \left(a^4 + a^3 + a + 1\right) x^{235} + \\
& \left(a^7 + a^4 + a^2 + a\right) x^{234} + \left(a^7 + a^6 + a^5 + a^4\right) x^{233} + \left(a^7 + a^6\right) x^{232} + \left(a^7 + a^6 + a^4 + a^3 + a\right) x^{231} \\
& + \left(a^7 + a^6 + a^5 + a^3 + a^2 + 1\right) x^{230} + \left(a^7 + a^6 + a^5 + a^4 + a^2 + a\right) x^{229} + \left(a^6 + a^5 + a\right) x^{228} + \\
& \left(a^7 + a^6 + a^5 + a^3\right) x^{227} + \left(a^7 + a^6 + a\right) x^{226} + \left(a^5 + a^3 + a^2\right) x^{225} + \\
& \left(a^7 + a^6 + a^5 + a^3 + a^2 + 1\right) x^{224} + \left(a^7 + a^5 + a^3 + a\right) x^{223} + \left(a^5 + a^3\right) x^{222} + \\
& \left(a^7 + a^6 + a^3 + a + 1\right) x^{221} + \left(a^5 + a^4 + a + 1\right) x^{220} + \left(a^5 + a^4 + a^2\right) x^{219} + \left(a^6 + a^3 + a^2\right) x^{218} + \\
& \left(a^6 + a^5 + a^4 + a^3 + a^2 + 1\right) x^{217} + \left(a^7 + a^5 + a^2 + 1\right) x^{216} + \left(a^7 + a^6 + a^4 + a^3 + 1\right) x^{215} + \\
& \left(a^6 + a^4 + a^3\right) x^{214} + \left(a^7 + a^6 + a^5 + a^4 + a^2\right) x^{213} + \left(a^6 + a^4 + a + 1\right) x^{212} + \left(a^5 + a\right) x^{211} + \\
& \left(a^5 + a^4\right) x^{210} + \left(a^7 + a^6 + a^5 + a^2 + a + 1\right) x^{209} + \left(a^7 + a^6 + a^5 + a^3\right) x^{208} +
\end{aligned}
$$

$$(a^5+a^4+a^3+a^2+a+1)x^{207}+(a^5+a^4+a^3+a^2+a)x^{206}+(a^6+a^5+a^4+a^3+1)x^{205}+$$
$$(a^7+a^4+a^3+1)x^{204}+(a^5+a^3+a)x^{203}+(a^7+a^4+a^3+a)x^{202}+(a^6+a^5+a^3)x^{201}+$$
$$(a^7+a^3+a^2+a+1)x^{200}+(a^5+a+1)x^{199}+(a^7+a^6+a^5)x^{198}+$$
$$(a^7+a^4+a^3+a^2+a+1)x^{197}+(a^7+a^6+a^5+a^4+a^3+a^2+a)x^{196}+(a^5+a^3)x^{195}+$$
$$(a^6+a^5+a^4+a^3+a)x^{194}+(a^6+a^2+a)x^{193}+(a^6+a^3+a+1)x^{192}+(a^4+1)x^{191}+$$
$$(a^4+a+1)x^{190}+(a^7+a^3+a^2+a)x^{189}+(a^6+a^5+a^4+a^3+a^2+a+1)x^{188}+$$
$$(a^7+a^6+a^5+a^2+a)x^{187}+(a^6+a^5+a^3+a)x^{186}+(a^5+a^4+a^2+1)x^{185}+a^7x^{184}+$$
$$(a^6+a^5+a^4+a^3+a^2+1)x^{183}+(a^7+a^6+a^5+a^3)x^{182}+(a^7+a^6+a^4+a^2)x^{181}+$$
$$(a^7+a^3+a^2+a+1)x^{180}+(a^7+a^2+a)x^{179}+a^7x^{178}+(a^7+a^6+a^4+a^3+a+1)x^{177}+$$
$$(a^6+a^5+a^3+a^2+a+1)x^{176}+(a^7+a^6+a^5+a^3+a^2+a+1)x^{175}+(a^3+1)x^{174}+$$
$$(a^7+a^6+a^3+a^2+1)x^{173}+a^6x^{172}+(a^7+a^4+1)x^{171}+a^2x^{169}+(a^7+1)x^{168}+$$
$$(a^7+a^4+a^2+a)x^{167}+ax^{166}+(a^7+a^4+a)x^{165}+(a^5+a^3+a)x^{164}+(a^6+a^4+a+1)x^{163}$$
$$+(a^7+a^5+1)x^{162}+(a^7+a^6+a^4+1)x^{161}+(a^6+a^5+a^2+a+1)x^{160}+$$
$$(a^6+a^5+a^3+a)x^{159}+(a^6+a^4+a^2)x^{158}+(a^7+a^4)x^{157}+(a^5+1)x^{156}+$$
$$(a^7+a^5+a^3+a^2+a+1)x^{155}+(a^7+a^5+a^3)x^{154}+(a^7+a^5+a^4+a^3)x^{153}+$$
$$(a^7+a^6+a^5+a)x^{152}+(a^4+a^3+a)x^{151}+(a^6+a^4+a^2+1)x^{150}+(a^3+a)x^{149}+$$
$$(a^5+a^4+a^3+a)x^{148}+(a^7+a^6+a^5+a^4+a^3+a^2+a)x^{147}+(a^7+a^5+a^3+a)x^{146}+$$
$$x^{145}+(a^6+a^5+a^4+a^3+a^2)x^{144}+ax^{143}+(a^7+a^6+a^3+a^2+1)x^{142}+$$
$$(a^7+a^6+a^4+a^2+a+1)x^{141}+(a^7+a^5+a^4+a^3+a+1)x^{140}+(a^6+a^5+a^3)x^{139}+$$
$$(a^6+a^5+a^4+a^2+a+1)x^{138}+(a^5+a^4+a^3+a)x^{137}+(a^7+a^6+a^2+1)x^{136}+$$
$$(a^5+a^4+a^3+1)x^{135}+(a^7+a^4+a^2)x^{134}+(a^7+a^6+a^4+a^3+a^2+a)x^{133}+(a^6+a)x^{132}$$
$$+(a^5+a^4+a^3+a^2+1)x^{131}+(a^5+a^2+a)x^{130}+(a^7+a^6+a^5+a^4+a^2+a+1)x^{129}+$$
$$(a^6+a^5+a^4+a^3)x^{128}+(a^7+a^6+a^4+a^3+a+1)x^{127}+(a^4+a^2+a+1)x^{126}+$$
$$(a^7+a^2+1)x^{125}+(a^6+1)x^{124}+(a^7+a^5+a^3+1)x^{123}+(a^7+a^6+a)x^{122}+$$
$$(a^7+a^5+a^3+a^2+1)x^{121}+(a^7+a^5+a^4+a^3+a^2+a)x^{120}+(a^7+a^6+a^4)x^{119}+$$
$$(a^5+a^4+a^3+a)x^{118}+(a^6+a^5+a^3+a^2+1)x^{117}+(a^5+a^4+a^2+1)x^{116}+$$
$$(a^7+a^5+a^3+a^2)x^{115}+(a^6+a^5+a^4+a^2+a)x^{114}+(a^6+a^5+a^4+a^2+a+1)x^{113}+$$
$$(a^7+a^6+a^3+a^2+a+1)x^{112}+(a^5+a^4+a^2+a+1)x^{111}+(a^7+a^6+a^4+a^3+a)x^{110}+$$
$$(a^6+a^5+a^4+1)x^{109}+(a^5+a^4+a^3+a^2+a+1)x^{108}+(a^7+a^6+a^5+a^4+a+1)x^{107}+$$
$$(a^6+a^5+a^2+a)x^{106}+(a^7+a^6+a^4)x^{105}+(a^7+a^5+a^3)x^{104}+(a^7+a^3+1)x^{103}+$$
$$(a^4+a)x^{102}+(a^7+a^6+a^5+a^3+1)x^{101}+(a^5+a^4+a^3)x^{100}+$$
$$(a^7+a^6+a^5+a^4+a+1)x^{99}+(a^7+a^5+a^3+a^2+a)x^{98}+(a^6+a^5+a^3+a^2+1)x^{97}+$$
$$(a^4+a^3+a+1)x^{96}+(a^4+a^3+a+1)x^{95}+(a^7+a^6+a^5+a^4+a^3+a)x^{94}+$$

$$(a^6+a^5+a^2+a)x^{93} + (a^6+a^5+a^4+a+1)x^{92} + (a^7+a^6+a^5+a)x^{91} + (a^6+a^4+a^3+1)x^{90} + (a^7+a^6+1)x^{89} + (a^7+a^6+a^5+a^4+a^2+a)x^{88} + (a^7+a)x^{87} + (a^6+a^5+a^4+a^3+a^2+a)x^{86} + (a^7+a^5+a^3+a^2+1)x^{85} + (a^7+a^2+a+1)x^{84} + (a^7+a^5+a^4+a)x^{83} + (a^6+1)x^{82} + x^{81} + (a^7+a^3+a^2+1)x^{80} + (a^7+a^6+a^4+a^2+1)x^{79} + (a^7+a^5+a^2)x^{78} + (a^5+a^3+a^2)x^{77} + (a^7+a^6+a^5+a^4+a^3+1)x^{76} + (a^7+a^6+a^4+a^3+1)x^{75} + (a^7+a^6+a^2+1)x^{74} + (a^6+a^2+1)x^{73} + (a^7+a^4+a^2)x^{72} + (a^6+a^3+a^2+a)x^{71} + (a^7+a^6+a^3+a)x^{70} + (a^7+a^5+a^4+a^3+a)x^{69} + (a^5+a^3+a+1)x^{68} + (a^5+a+1)x^{67} + (a^2+a)x^{66} + (a^6+a^5+a^3+1)x^{65} + (a^7+a^6+a^5+a^4+a^3+a^2+1)x^{64} + (a^6+a^4+a+1)x^{63} + (a^5+a^2+1)x^{62} + (a^6+a^5+a^4+a^2)x^{61} + (a^4+a^3)x^{60} + (a^6+a^3+a^2+a+1)x^{59} + (a^7+a^4+a^3+a^2+a+1)x^{58} + (a^7+a^6+a^5+a^2+a+1)x^{57} + (a^5+a^4+a^3+a^2)x^{56} + (a^3+1)x^{55} + (a^7+a^6+a^5+a^4+a^3)x^{54} + (a^7+a^4+a+1)x^{53} + (a^7+a^6+a^5+a^2+a+1)x^{52} + (a^6+a^4+a+1)x^{51} + (a^7+a^6+a^5+a^4+a^3+a)x^{50} + (a^7+a^6+a^5+a^3+a^2+a+1)x^{49} + (a^7+a^6+a^4+a^2)x^{48} + (a^7+a^6+a^5+a^4+a^3+1)x^{47} + (a^5+a^3+a^2+1)x^{46} + (a^5+a^3+a)x^{45} + (a^7+a^3)x^{44} + (a^6+a^5+a^4+a^3+a+1)x^{43} + (a^7+a^6+a^5+a^4+a^2)x^{41} + (a^6+a^5+a^3+1)x^{40} + (a^6+a^5+a^3+a^2+1)x^{39} + (a^7+a^5+a^3+a^2+1)x^{38} + (a^4+a^3+a^2+a)x^{37} + (a^7+a^6+a^3+a+1)x^{36} + (a^7+a^6+a^5+a)x^{35} + (a^7+a^5+a^3)x^{34} + (a^7+a^6+a^4+a^3+a)x^{33} + (a^7+a^6+a^5+a^4+a^3+a^2)x^{32} + (a^6+a^3+a^2+a)x^{31} + (a^7+a^6+a^4+a^3+a^2+1)x^{30} + (a^6+a^5+a^2+a)x^{29} + (a^6+a^5+a^4+a^3+a^2+a)x^{28} + (a^5+a^3+a^2)x^{27} + (a^5+a^4+a^3+a+1)x^{26} + (a^7+a^3)x^{25} + (a^7+a^4+a^2+1)x^{24} + (a^3+a^2+1)x^{23} + (a^7+a^6+a)x^{22} + (a^6+a^2+1)x^{21} + (a^7+a^3+1)x^{20} + (a^4+1)x^{19} + (a^7+a^4+a^3+1)x^{18} + (a^7+a^6+a^5+a^2+a+1)x^{17} + (a^5+a^3+1)x^{16} + (a^5+a^4+a+1)x^{15} + (a^7+a^5+a^4+a^3+a^2+1)x^{14} + (a^5+a^3+a^2+a)x^{13} + (a^5+a^3+a^2)x^{12} + (a^7+a^5+a^3+a)x^{11} + (a^5+a^4+a^3+a)x^{10} + (a^4+a^3+a^2+a+1)x^9 + (a^6+a^5+a^4)x^8 + (a^7+a^5+a^4+a^2+a)x^7 + (a^5+a^4+a^3+a^2+1)x^6 + (a^7+a^4+1)x^5 + (a^6+a^5+a^3+a^2)x^4 + (a^6+a^4+a^3+a^2)x^3 + (a^6+a^5)x^2 + (a^4+a^3)x + a^6 + a^5 + a + 1$$

$$S^{-1}(x) = (a+1)x^{254} + (a^7+a^2+a)x^{253} + (a^6+a^4+a^3+a+1)x^{252} + (a^6+a^3+a^2+1)x^{251} + (a^5+a^3+a^2+1)x^{250} + (a^6+a^5+a^4+a+1)x^{249} + (a^7+a)x^{248} + (a^7+a^5+a^2+1)x^{247} + (a^7+a^6+a^5+a^2+1)x^{246} + (a^7+a^5+a^4+a^2+a+1)x^{245} + (a^7+a^5+a^4+a^3+a^2)x^{244} + (a^6+a^5+a^4+a^3+a^2+a+1)x^{243} + (a^7+a^6+a^4+a^2)x^{242} + (a^5+a^4+a^2)x^{241} + (a^7+a^4+a^2)x^{240} + (a^7+a^6+1)x^{239} +$$

$$(a^4+a^3+a+1)x^{238} + (a^6+1)x^{237} + (a^7+a^6+a^3+a)x^{236} +$$
$$(a^7+a^6+a^5+a^4+a)x^{235} + (a^6+a^3+1)x^{234} + (a^6+a^4+a^2+a)x^{233} +$$
$$(a^7+a^4+a^3)x^{232} + (a^7+a^6+a^5+a^4+a^2+a)x^{231} + (a^7+a^6+a+1)x^{230} +$$
$$(a^7+a^3+a^2+a)x^{229} + (a^7+a^4+a)x^{228} + (a^7+a^6+a^4+a^2)x^{227} + a^5x^{226} +$$
$$(a^5+a^3+a^2+a)x^{225} + (a^4+a^2+a)x^{224} + (a^7+a^6+a^5+a^3+a+1)x^{223} +$$
$$(a^6+a^2+a+1)x^{222} + (a^6+a^4+a^2)x^{221} + (a^7+a^6+a^5+a^4+1)x^{220} +$$
$$(a^6+a^5+a^4+a^3+a^2+1)x^{219} + (a^5+a^3)x^{218} + (a^7+a^6+a^3+a^2+a+1)x^{217} +$$
$$(a^5+a^4+a^2)x^{216} + (a^6+a^5+a^4+a^3)x^{215} + (a^7+a^6+a^5+a^3)x^{214} +$$
$$(a^7+a^6+a^5+a^3+a+1)x^{213} + (a^3+a+1)x^{212} + (a^4+a^3+a^2)x^{211} + (a^4+1)x^{210} +$$
$$(a^7+a^3+a^2)x^{209} + (a^6+a^5+a^2+1)x^{208} + (a^5+a^3+a^2)x^{207} + (a^5+a^3+a^2+a)x^{206} +$$
$$(a^7+a^6+a^5+a^3+a^2)x^{205} + (a^7+a^4+a)x^{204} + (a^6+a^5+a^4)x^{203} + (a^5+a^4+a^2)x^{202} +$$
$$(a^5+a^3+1)x^{201} + (a^7+a^6+a^4+a^3+a^2+a)x^{200} + (a^7+a^5+a^4+a^3+1)x^{199} +$$
$$(a^7+a^6+1)x^{198} + (a^7+a^4+1)x^{197} + (a^7+a^4+a+1)x^{196} + (a^7+a^4+a^3+a^2+1)x^{195} +$$
$$(a^7+a^6+a^4+a^3+a^2)x^{194} + (a^7+a^6+a^5+a^2+a)x^{193} + (a^6+a^5+a^3+1)x^{192} +$$
$$(a^6+a^5+a^4+a^3+a+1)x^{191} + (a^7+1)x^{190} + (a^7+a^4+a^2+a)x^{189} + (a^5+a^2+1)x^{188} +$$
$$(a^7+a^6+a^5+a^2+a)x^{187} + (a^5+a^4)x^{186} + (a^6+a^3+a^2+a+1)x^{185} +$$
$$(a^7+a^6+a^4+a^3+a^2)x^{184} + (a^7+a^5+1)x^{183} + (a^7+a^4+a^3+a^2+1)x^{182} +$$
$$(a^7+a^5+a^4+1)x^{181} + (a^7+a^6+a^3+a^2+a+1)x^{180} + (a^6+a^3+a^2+a+1)x^{179} +$$
$$(a^5+a^3+1)x^{178} + (a^7+a^6+a^5+a^4+a^3+1)x^{177} + (a^6+a^4+a)x^{176} + (a^6+a^5+a)x^{175}$$
$$+ (a^7+a^4+a^3+a^2+a+1)x^{174} + (a^7+a^6+a^4+a^3+a^2+a)x^{173} +$$
$$(a^7+a^6+a^5+a^4+a+1)x^{172} + (a^6+a^5+a^4+a)x^{171} + (a^5+1)x^{170} +$$
$$(a^7+a^2+a+1)x^{169} + (a^7+a^5+a^4+a^3+a)x^{168} + (a^6+a^4+a^3+a)x^{167} +$$
$$(a^5+a^4+1)x^{166} + (a^6+a^5+a^2+a)x^{165} + (a^4+a^2)x^{164} + (a^7+a^3+a+1)x^{163} +$$
$$(a^4+a^2+1)x^{162} + (a^7+a^6+a^3+a^2+1)x^{161} + (a^6+a^3+a^2+a+1)x^{160} +$$
$$(a^4+a^2+1)x^{159} + (a^6+a^3+a)x^{158} + (a^5+a^2+a+1)x^{157} + (a^6+a^5+a^3+a)x^{156} +$$
$$(a^6+a^5+a^2+1)x^{155} + (a^7+a^4+a^3+1)x^{154} + (a^7+a^3)x^{153} + (a^7+a^6+a^2)x^{152} +$$
$$(a^7+a^6+a^5+a^3+a^2+a)x^{151} + (a^7+a^6+a^2+a)x^{150} + (a^7+a^5)x^{149} +$$
$$(a^7+a^5+a^4+a^3+a^2+a)x^{148} + (a^7+a^3+a^2+a)x^{147} + (a^7+a^5+a^3+1)x^{146} +$$
$$(a^6+a^5+a^4+a^3+a^2)x^{145} + (a^5+a^4+a+1)x^{144} + (a^5+a^4+a)x^{143} +$$
$$(a^7+a^6+a^2+1)x^{142} + (a^7+a^6+a^5+a^4+a^2+a)x^{141} + (a^6+a^5+a^4+a+1)x^{140} +$$
$$(a^7+a^5+a^4+a^3+a^2+1)x^{139} + (a^7+a^5+a^4+a^3+a^2+a+1)x^{138} +$$
$$(a^7+a^5+a^3+a^2+1)x^{137} + (a^7+a^4+a^3+a^2)x^{136} + (a^7+a^6+a^5+a^4+a^3)x^{135} +$$
$$(a^7+a^6+a^5+a^4)x^{134} + (a^4+a^3+1)x^{133} + (a^7+a^5+a^4+a^3)x^{132} +$$

$$(a^5+a^4+a^3+a^2)x^{131} + (a^7+a^4+a^2+a+1)x^{130} + (a^7+a^5+a^3+a+1)x^{129} +$$
$$(a^7+a^5+a^3+a^2+1)x^{128} + (a^5+a^2)x^{127} + (a^7+a^6+a^5+a^3+a^2+a+1)x^{126} +$$
$$(a^7+a^6+a^5+a^4+a^3+a)x^{125} + (a^6+a^3)x^{124} + (a^7+a^5+a^2+a+1)x^{123} +$$
$$(a^7+a^6+a^5+a^4+a^3+a^2+a)x^{122} + (a^7+a^5+a^4+a+1)x^{121} +$$
$$(a^7+a^6+a^3+a^2+a+1)x^{120} + (a^3+1)x^{119} + (a^7+a^3+a^2)x^{118} +$$
$$(a^7+a^6+a^4+a^3+a^2+a)x^{117} + (a^5+a^2+1)x^{116} + (a^6+a^4+a^2)x^{115} +$$
$$(a^6+a^5+a^3+1)x^{114} + (a^4+a^2+a+1)x^{113} + (a^6+a^5+a^2+a+1)x^{112} +$$
$$(a^5+a^4+a^3+a^2+1)x^{111} + (a^7+a^6+a^4+a^3+a)x^{110} + (a^6+a^5+a^3+a^2+a+1)x^{109}$$
$$+ (a^7+a^6+a^5+a^3+a)x^{108} + (a^3+a+1)x^{107} + (a^7+a^5+1)x^{106} +$$
$$(a^6+a^5+a^4+a^3)x^{105} + (a^7+a^6+a^5+1)x^{104} + (a^6+a^5+a)x^{103} + (a^7+a^3+1)x^{102} +$$
$$(a^6+a^5+a^4+a^3+a)x^{101} + (a^7+a^5+a^4+a^3+1)x^{100} + (a^7+a^6+a^4+a^3+a^2)x^{99} +$$
$$a^5x^{98} + (a^7+a^6+a+1)x^{97} + (a^6+a^4+a^2)x^{96} + (a^7+a^5+a^3+a+1)x^{95} +$$
$$(a^7+a^5+a^3+a^2)x^{94} + (a^7+a^6+a^3+a^2+1)x^{93} + (a^6+a^4+a^3+a^2+1)x^{92} +$$
$$(a^7+a^5+a^2+a+1)x^{91} + (a^6+a^4+a^3+a+1)x^{90} + (a^7+a^6+a^5+a^4+a^2+a)x^{89} +$$
$$(a^5+a^4+a)x^{88} + (a^7+a^6+a^5+a^3+a)x^{87} + (a^7+a^6+a^5+a+1)x^{86} +$$
$$(a^6+a^3+1)x^{85} + (a^6+a^3+a^2+a+1)x^{84} + (a^6+a^3+a^2+a+1)x^{83} + (a^6+a^2)x^{82} +$$
$$(a^7+a^6+a^5+a^4+a^3+a^2+a+1)x^{81} + (a^6+a^5+a^4+1)x^{80} +$$
$$(a^7+a^6+a^4+a^3+a+1)x^{79} + a^4x^{78} + (a^6+a^4+a)x^{77} + (a^7+a^6+a^5+a^2+1)x^{76} +$$
$$(a^6+a^5+a^3+a)x^{75} + a^5x^{74} + (a^6+a^3+a+1)x^{73} + (a^7+a^5+a^4+a^2+a)x^{72} + a^6x^{71} +$$
$$(a^5+a^2)x^{70} + (a^7+a^6+a^5+a^4+1)x^{69} + (a^7+a^5+a^4+a^3+a+1)x^{68} + a^2x^{67} +$$
$$(a^6+a^5+a+1)x^{66} + (a^7+a^3+1)x^{65} + (a^7+a^6+a^4+a^3)x^{64} + (a^7+a^4+a^2+1)x^{63} +$$
$$(a^7+a^5+a^2+a+1)x^{62} + (a^6+a^5+a^4+a+1)x^{61} + (a^5+a^4+a^3+a)x^{60} +$$
$$(a^5+a^2+a)x^{59} + (a^7+a^5+a^3+a+1)x^{58} + (a^7+a^6+a^4+1)x^{57} +$$
$$(a^6+a^5+a^2+a)x^{56} + (a^2+1)x^{55} + (a^7+a^6+a+1)x^{54} + (a^6+a^3+a+1)x^{53} +$$
$$(a^6+a^3+a)x^{52} + (a^6+a^4+1)x^{51} + (a^4+a+1)x^{50} + (a^7+a^3+a+1)x^{49} +$$
$$(a^5+a^4+a^3+a^2+a+1)x^{48} + (a^7+a^5+a^3+a^2+1)x^{47} + (a^7+a^5+a^2+1)x^{46} +$$
$$(a^5+a^4+a^3+a^2+1)x^{45} + (a^7+a^4+a^3+a)x^{44} + (a^5+a^3+a^2)x^{43} + (a^7+a^6+1)x^{42} +$$
$$(a^7+a^5+a^3+a+1)x^{41} + (a^7+a^5+a^4+a^3+1)x^{40} + (a^4+a^3+a^2+a)x^{39} +$$
$$(a^7+a^5+a^3+a)x^{38} + (a^7+a^2+1)x^{37} + (a^7+a^6+a^4)x^{36} + (a^7+a^5+a^3+a^2+a)x^{35} +$$
$$(a^7+a^6+a^5+a^3+a^2)x^{34} + (a^7+a^6+a^5+a^4+a^2+a+1)x^{33} +$$
$$(a^7+a^6+a^4+a^3+a^2+a)x^{32} + (a^6+a^4+a^3+a^2+1)x^{31} + (a^7+a^5+1)x^{30} +$$
$$(a^7+a^6+a^2)x^{29} + (a^5+a^3+a^2+a+1)x^{28} + (a^5+a+1)x^{27} + (a^7+a^3+a^2+1)x^{26} +$$
$$(a^5+1)x^{25} + (a^7+a^6+a^5+a^4+a^3+a)x^{24} + (a^6+a^4+1)x^{23} + (a^7+a^4+a^2+1)x^{22} +$$

$$\left(a^6+a^5+a^4+a^3+a\right)x^{21} + \left(a^7+a^4+a^2+a\right)x^{20} + \left(a^7+a^6+a^4+a\right)x^{19} + \left(a^7+a^6+a^5+a^2+1\right)x^{18} + \left(a^7+a^3+a^2+a+1\right)x^{17} + \left(a^4+a^3+a^2\right)x^{16} + \left(a^6+a^5+a^4+a^3+a^2+1\right)x^{15} + \left(a^7+a^6+a^3+a^2+a+1\right)x^{14} + \left(a^7+a^6+a^3+a\right)x^{13} + \left(a^7+a^6+a^5\right)x^{12} + \left(a^7+a^4+1\right)x^{11} + \left(a^7+a^5+a^4+a\right)x^{10} + \left(a^7+a^6+a^4+a^3+a^2+1\right)x^9 + \left(a^7+a^6+a^4+a^2+a+1\right)x^8 + \left(a^7+a^6+a^4+a^2+a\right)x^7 + \left(a^7+a^5+a^4+a\right)x^6 + \left(a^7+a^6+a^5+a^2+a+1\right)x^5 + \left(a^6+a^5+a^3+a^2+a+1\right)x^4 + \left(a^4+a^3+a\right)x^3 + \left(a^7+a^4+a^3+a^2+a\right)x^2 + \left(a^6+a^3+a\right)x + a^6+a^4+a$$

A.2 SageMath Code

In this section we document some of the code used for perfect c-nonlinearity, c-differential bentness, and c-differential uniformity computations. Thank you to Professor Dibyendu Roy for greatly improving the code with the inclusion of multiprocessing.

A.2.1 c-Differential Bentness and Perfect c-Nonlinearity

In Section 3.4 we provided examples of PcN and c-differential bent functions by calculating their c-autocorrelations for various values of c. Recall a function is PcN (equivalently c-differential bent) if the non-trivial c-autocorrelation values are 0. For a specific c, we used the following code to compute the c-autocorrelation values:

```
from collections import Counter

#define the environment
n = (#choose n)
p = (#choose p)
c = (#choose c)
F = GF(p)
FF.<g> = GF(p^n)
FFlist = list(FF)
#non-zero finite field elements
FFstar = [i for i in FF][1:]
R.<x> = FF[]
U=list(FF)
```

```
#finite field without identity (1)
FF1=Set(U).difference([g^0])
#we need the primitive root for c-autocorrelation
z=exp(2*pi*I/p)

def trace(x):
    return sum(x^(p^s) for s in [0..n-1])

def f(x):
    return(#choose function to analyze)

#c-autocorrelation function
def cCb(f,c,a,b):
    return sum(z^(ZZ(trace(b*(f(x+a)-c*f(x))))) for x in FFlist)

#compute the c-autocorrelation table
auto=[[cCb(f,c,a,b) for b in FFlist] for a in FFlist]
#the "auto" will need to be formatted for easier table viewing
```

To find all values of c for which a function is PcN (i.e., c-differential bent), we added multiprocessing techniques and used the following:

```
import multiprocessing
from multiprocessing import Process
from collections import Counter

#define the environment
n = (#choose n)
p = (#choose p)
F = GF(p)
FF.<g> = GF(p^n)
FFlist = list(FF)
```

```
R.<x> = FF[]
U=list(FF)
FF1=Set(U).difference([g^0])

z=exp(2*pi*I/p)

def trace(x):
    return sum(x^(p^s) for s in [0..n-1])

def f(x):
    return(#enter function to analyze here)

#c-autocorrelation
def cCb(f,c,a,b):
    return sum(z^(ZZ(trace(b*(f(x+a)-c*f(x))))) for x in FFlist)

#nth is number of processes for multiprocessing
nth = 4
sz = p^n/nth

#Split the FF into 'nth' many rows for multiprocessing
FFth = matrix(FF,nth,sz,[U[i*sz:(i+1)*sz] for i in range(nth)])

Stt=[0]*nth
dfres=[0]*nth

#define a global list to merge from each process
manager = multiprocessing.Manager()
fl = manager.list()

#define function for each process
def thfun(num):
```

```
print('process id:', os.getpid())
    #following loop is for each row of FFth
    St=[]
    Stt1=[]
    for c in FFth[num]:
        St=[]
        Stt1=[]
        Su=[[simplify(expand(cCb(f,c,a,b))) for a in FFlist] for b in FFstar]
        #looking for all non-trivial values to be 0
        if set(flatten(simplify(Su)))=={0}:
            print(c)
            #prints values of c for PcN
```

A.2.2 c-Differential Uniformity

We first computed a function's cDU in Subsection 4.2.1. Later, in Chapter 6 we computed the cDU of the S-boxes of many real-world ciphers. The following code computes the maximum cDU of a function across all $c \neq 1$:

```
import multiprocessing
from multiprocessing import Process
from collections import Counter

#define the environment
n = (#choose n)
p = (#choose p)
F = GF(p)
FF.<g> = GF(p^n)
FFlist = list(FF)

R.<x> = FF[]
U=list(FF)
```

```
FF1=Set(U).difference([g^0])

def f(x):
    return(#enter function to analyze here)

#nth is number of processes for multiprocessing
nth = 4
sz = p^n/nth

#Split the FF into 'nth' many rows for multiprocessing
FFth = matrix(FF,nth,sz,[U[i*sz:(i+1)*sz] for i in range(nth)])

Stt=[0]*nth
dfres=[0]*nth

#define a global list to merge from each process
manager = multiprocessing.Manager()
fl = manager.list()

#define function for each process
def thfun(num):
    print('process id:', os.getpid())
    #following loop is for each row of FFth
    for a in FFth[num]:
     #The c-derivative counts, running through c not equal to 1
        dfres[num] = [[f(x+a) - c*f(x) for x in FF] for c in FF1]
        freq=[]
        for i in range(len(dfres[num])):
            freq.append(Counter(dfres[num][i]).most_common())
        St=[]
        Stt1=[]
        for i in [0..len(freq)-1]:
            for j in [0..len(freq[i])-1]:
```

```
                St.append(freq[i][j][1])
    Stt1.append(list(flatten(list(St))))
    fl.append(St)

# defining jobs
jobs = []
for i in range(0, nth):
    out_list = list()
    pro = Process(target=thfun, args=(i,))
    jobs.append(pro)

#starting each process
for j in jobs:
    j.start()
#joining each process
for j in jobs:
    j.join()

print('process complete')
print('Max Difference: ',max(flatten(fl)))
#max difference is the cDU across all c not equal to 1
```

THIS PAGE INTENTIONALLY LEFT BLANK

List of References

[1] C. Shannon, "Communication theory of secrecy systems," *Bell Systems Technical Journal*, vol. 28, pp. 656–715, 1949.

[2] P. Ellingsen, P. Felke, C. Riera, P. Stănică, and A. Tkachenko, "C-differentials, multiplicative uniformity and (almost) perfect c-nonlinearity," *IEEE Transactions on Information Theory*, vol. 66, no. 9, pp. 5781–5789, 2020.

[3] T. W. Cusick and P. Stănică, *Cryptographic Boolean Functions and Applications*, 2nd ed. Academic Press, 2017.

[4] L. Budaghyan, *Construction and Analysis of Cryptographic Functions*, 1st ed. Springer, 2015.

[5] S. Mesnager, *Bent Functions: Fundamentals and Results*, 1st ed. Springer, 2016.

[6] N. Tokareva, *Bent Functions: Results and Applications to Cryptography*, 1st ed. Academic Press, 2015.

[7] C. Carlet, "Boolean functions for cryptography and error correcting codes," in *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*, Y. Crama and P. L. Hammer, Eds., 1st ed. Cambridge Univ. Press, 2010, ch. 8, pp. 257–397.

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

[9] A. Canteaut, "Lecture notes on cryptographic Boolean functions," *Inria, Paris, France*, 2016.

[10] K. Nyberg, "Perfect nonlinear S-boxes," in *Advances in Cryptology – EUROCRYPT '91*. Springer, 1992, pp. 378—386.

[11] W. Meier and O. Staffelbach, "Nonlinearity criteria for cryptographic functions," in *Advances in Cryptology — EUROCRYPT '89*, J. Quisquater and J. Vandewalle, Eds. Springer, 1990, pp. 549–562.

[12] C. Carlet, "Vectorial Boolean functions for cryptography," in *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*, Y. Crama and P. L. Hammer, Eds., 1st ed. Cambridge Univ. Press, 2010, ch. 9, pp. 398–472.

[13] A. Canteaut and L. Perrin, "On CCZ-equivalence, extended-affine equivalence, and function twisting," *Finite Fields and Their Applications*, vol. 56, pp. 209–246, 2019.

[14] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," *Journal of CRYPTOLOGY*, vol. 4, no. 1, pp. 3–72, 1991.

[15] D. Coppersmith, "The Data Encryption Standard (DES) and its strength against attacks," *IBM Journal of Research and Development*, vol. 38, pp. 243–250, 1994.

[16] "Announcing Development of a Federal Information Processing Standard for Advanced Encryption Standard," Federal Register, Vol. 62, No. 1, National Institute of Standards and Technology (NIST), Department of Commerce, January 1997.

[17] H. M. Heys, "A tutorial on linear and differential cryptanalysis," *Cryptologia*, vol. 26, no. 3, pp. 189–221, 2002.

[18] K. Nyberg and L. Knudsen, "Provable security against differential cryptanalysis," in *Advances in Cryptology – CRYPTO '92*. Springer, 1992, pp. 566–574.

[19] L. Knudsen, "Truncated and higher order differentials," in *Fast Software Encryption*, B. Preneel, Ed. Springer, 1994, pp. 196–211.

[20] X. Lai, "Higher order derivatives and differential cryptanalysis," in *Communications and Cryptography*. Springer, 1994, pp. 227–233.

[21] L. Knudsen and T. Jakobsen, "The interpolation attack on block ciphers," in *Fast Software Encryption*, B. Preneel, Ed. Springer, 1997, pp. 28–40.

[22] E. Biham, A. Biryukov, and A. Shamiri, "Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials," in *Advances in Cryptology – EUROCRYPT '99*, J. Stern, Ed. Springer, 1999, pp. 12–23.

[23] D. Wagner, "The boomerang attack," in *Fast Software Encryption*, L. Knudsen, Ed. Springer, 1999, pp. 156–170.

[24] N. Borisov, M. Chew, R. Johnson, and D. Wagner, "Multiplicative differentials," in *Fast Software Encryption*, J. Daemen and V. Rijmen, Eds. Springer, 2002, pp. 17–33.

[25] P. Stănică, S. Gangopadhyay, A. Geary, C. Riera, and A. Tkachenko, "C-differential bent functions and perfect nonlinearity," *Discrete Applied Mathematics*, vol. 307, pp. 160–171, 2022.

[26] R. Lidl and H. Niederreiter, *Finite Fields*, 2nd ed. Cambridge Univ. Press, 1997.

[27] D. Bartoli and M. Calderini, "On construction and (non)existence of c-(almost) perfect nonlinear functions," *Finite Fields and Their Applications*, vol. 72, p. 101835, 2021.

[28] P. Stănică, T. Martinsen, S. Gangopadhyay, and B. K. Singh, "Bent and generalized bent Boolean functions," *Designs, Codes and Cryptography*, vol. 69, no. 1, pp. 77–94, 2013.

[29] P. Ellingsen, C. Riera, P. Stănică, and A. Tkachenko, "An extension of the avalanche criterion in the context of c-differentials," *The 18th International Conference on Security and Cryptography (SECRYPT 2021)*, pp. 460–467, 2021.

[30] X. Zhang and Y. Zheng, "GAC - the criterion for global avalanche characteristics of cryptographic functions," *Journal of Universal Computer Science*, vol. 1, pp. 320–337, 1996.

[31] *Announcing the ADVANCED ENCRYPTION STANDARD (AES)*, Federal Information Processing Standards Publication 197, November 26 2001.

[32] J. B. Fraleigh, *A First Course in Abstract Algebra*. Pearson Education, 2003.

[33] S. U. Hasan, M. Pal, C. Riera, and P. Stănică, "On the c-differential uniformity of certain maps over finite fields," *Designs, Codes and Cryptography*, vol. 89, no. 2, pp. 221–239, 2021.

[34] P. Stănică and A. Geary, "The c-differential behavior of the inverse function under the EA-equivalence," *Cryptography and Communications*, vol. 13, no. 2, pp. 295–306, 2021.

[35] N. Bourbaki, *Elements of Mathematics, Algebra II*. Springer, 1990 (translated by P. M. Cohn and J. Howie).

[36] E. Berlekamp, H. Rumsey, and G. Solomon, "On the solutions of algebraic equations over finite fields," *Information and Control*, vol. 10, pp. 553–564, 1967.

[37] K. Williams, "Note on cubics over $GF(2^n)$ and $GF(3^n)$," *Journal of Number Theory*, vol. 7, no. 4, pp. 361–365, 1975.

[38] R. S. Coulter, "Explicit evaluations of some Weil sums," *Acta Arithmetica*, vol. 83, no. 3, pp. 241–251, 1998.

[39] A. W. Bluher, "On $x^{q+1}+ax+b$," *Finite Fields and Their Applications*, vol. 10, no. 3, pp. 285–305, 2004.

[40] T. Helleseth and A. Kholosha, "On the equation $x^{2^l+1} + x + a$ over $GF(2^k)$," *Finite Fields and Their Applications*, vol. 14, pp. 159–176, 2008.

[41] D. Mills, "On the evaluation of Weil sums of Dembowski–Ostrom polynomials," *Journal of Number Theory*, vol. 92, pp. 87–98, 2002.

[42] P. Stănică, "Using double Weil sums in finding the c-Boomerang Connectivity Table for monomial functions on finite fields," *Applicable Algebra in Engineering, Communication and Computing*, pp. 1–22, 2021.

[43] Z. Zha and L. Hu, "Some classes of power functions with low c-differential uniformity over finite fields," *Designs, Codes and Cryptography*, vol. 89, no. 6, pp. 1193–1210, 2021.

[44] X. Wang and D. Zheng, "Several classes of PcN power functions over finite fields," 2021.

[45] S. Mesnager, C. Riera, P. Stănică, H. Yan, and Z. Zhou, "Investigations on c-(almost) perfect nonlinear functions," *IEEE Transactions on Information Theory*, vol. 67, no. 10, pp. 6916–6925, 2021.

[46] H. Yan, "On -1-differential uniformity of ternary APN power functions," 2021.

[47] P. Stănică, "Low c-differential uniformity for the Gold function modified on a subfield," *Security and Privacy: Select Proceedings of ICSP 2020*, vol. Springer, LNEE 744, pp. 131–137, 2020.

[48] Y. Wu, N. Li, and X. Zeng, "New PcN and APcN functions over finite fields," *Designs, Codes and Cryptography*, pp. 1–15, 2021.

[49] A. Geary, M. Calderini, C. Riera, and P. Stănică, "Higher order c-differentials," in *International Conference on Security and Privacy*. Springer, 2021, pp. 3–15.

[50] D. Tang, B. Mandal, and S. Maitra, "Further cryptographic properties of the multiplicative inverse function." *IACR Cryptol. ePrint Arch.*, vol. 2020, p. 920, 2020.

[51] A. Bogdanov, L. Knudsen, G. Leander, C. Paar, A. Poschmann, M. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: An ultra-lightweight block cipher," in *Cryptographic Hardware and Embedded Systems – CHES 2007*, P. Paillier and I. Verbauwhede, Eds. Springer, 2007, pp. 450–466.

[52] W. Zhang, Z. Bao, D. Lin, V. Rijmen, B. Yang, and I. Verbauwhede, "RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms," *Science China Information Sciences*, vol. 58, pp. 1–15, 2015.

[53] E. Biham, R. Anderson, and L. Knudsen, "Serpent: A new block cipher proposal," in *Fast Software Encryption*. Springer, 1998, pp. 222–238.

[54] T. Shimoyama, H. Yanami, K. Yokoyama, M. Takenaka, K. Itoh, and J. Yajima, "The block cipher SC2000," in *Fast Software Encryption*. Springer, 2001, pp. 312–327.

[55] J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knezevic, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, P. Rombouts, S. S. Thomsen, and T. Yalçın, "Prince – a low-latency block cipher for pervasive computing applications," in *Advances in Cryptology – ASIACRYPT 2012*, X. Wang and K. Sako, Eds. Springer, 2012, pp. 208–225.

[56] B. Bilgin, A. Bogdanov, M. Knežević, F. Mendel, and Q. Wang, "Fides: Lightweight authenticated cipher with side-channel resistance for constrained hardware," in *Cryptographic Hardware and Embedded Systems – CHES 2013*, G. Bertoni and J. Coron, Eds. Springer, 2013, pp. 142–158.

[57] S. Riour, "DryGASCON," in *Lightweight Cryptography Standardization Process Round 1 Submission. NIST*, 2019.

[58] D. Penazzi and M. Montes, "Shamash (and shamashash) (version 1)," in *Lightweight Cryptography Standardization Process round 1 submission. NIST*, 2019.

[59] B. Kelsey, D. Whiting, D. Wagner, C. Hall, and N. Ferguson, "Twofish: A 128-bit block cipher," 1998.

[60] T. Shirai, K. Shibutani, T. Akishita, S. Moriai, and T. Iwata, "The 128-bit blockcipher CLEFIA," in *Fast Software Encryption*. Springer, 2007, pp. 181–195.

[61] L. Shuwang, "The SM4 blockcipher algorithm and its modes of operations," 2016.

[62] M. Calderini, M. Sala, and I. Villa, "A note on APN permutations in even dimension," *Finite Fields and Their Applications*, vol. 46, pp. 1–16, 2017.

[63] K. Browning, J. Dillon, M. McQuistan, and A. Wolfe, "An APN permutation in dimension six," *Finite Fields: Theory and Applications*, vol. 518, pp. 33–42, 2010.

THIS PAGE INTENTIONALLY LEFT BLANK

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California