Cryptanalytic Properties of Mealy Machines*

Zhongfeng Niu[2][0009−0009−6932−2116], Tim Beyne[3][0000−0001−5638−9885], Kai Hu[1,4,5][0000−0003−3552−7200](✉), and Meiqin Wang[1,6][0000−0003−1580−6544]

[1] School of Cyber Science and Technology, Shandong University, Qingdao, China
kai.hu@sdu.edu.cn, mqwang@sdu.edu.cn,
[2] School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore
zhongfeng.niu@ntu.edu.sg
[3] COSIC, KU Leuven, Leuven, Belgium
tim.beyne@esat.kuleuven.be
[4] State Key Laboratory of Cryptography and Digital Economy Security, Shandong University, Qingdao, China
[5] Suzhou Research Institute, Shandong University, Suzhou, 215123, China
[6] Quan Cheng Shandong Laboratory, Jinan, China

Abstract. This paper proposes a systematic approach to compute cryptanalytic properties of arbitrary Mealy machines or *S-functions*. Based on the geometric approach to cryptanalysis, we provide a uniform formula for any cryptanalytic property of such a function, as long as the property is compatible with the way its input and output are split into chunks. Examples include linear, (quasi) differential, (ultrametric) integral, differential-linear, and boomerang properties. To illustrate our results, we compute these properties for several important examples, including modular additions, the χ- and $\chi\!\!\!\chi$-functions, and the SHA-1 step function. As proof-of-concept applications, we construct a boomerang distinguisher for the Subterranean 2.0 permutation, and show how to compute the correlations of conditional linear approximations in partitioning-based differential-linear attacks more accurately. Our results also lead to a new approach to compute the algebraic normal form of the inverse of the χ-function.

Keywords: S-functions · Cryptanalysis · Geometric approach

1 Introduction

Most symmetric-key primitives are constructed by composing many operations that are individually simple. These operations have to be efficient, but they also need to operate on a relatively large state (in typical cases, at least 128 bits).

For nonlinear operations, this is commonly achieved by producing output bits sequentially based on small chunks of the input, and on a constant-size state that depends on earlier input chunks. This model of computation, illustrated in Figure 1 on the next page, is known as a *Mealy machine* [34]. In the most

* The first three authors contributed equally.

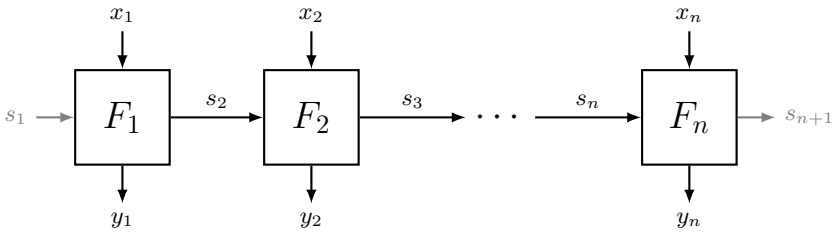


Fig. 1: Circuit representation of a Mealy machine or 'S-function'.

extreme case, the state is empty, and the operation reduces to a layer of parallel S-boxes. However, other Mealy machines (such as modular addition) are also widely used as building blocks of symmetric-key primitives.

In the cryptographic literature, the functions computed by Mealy machines have been called *S-functions* [37]. S-functions are closely related to T-functions, which were introduced earlier by Klimov and Shamir [27]. In a T-function, the i^{th} output chunk only depends on the first i input chunks, but unlike for S-functions, they need not be computable using a constant-size state. However, many T-functions that are used in practice are also S-functions.

Surprisingly many building blocks of symmetric-key primitives are S-functions. Important examples include the modular addition between two or more addends, modular addition with a constant, and Daemen's χ-function[7] [18]. For some primitives, even the entire round function is an S-function — up to inconsequential changes such as reordering the input and output chunks. Examples include the step functions of hash functions such as MD-5 and SHA-1, and the round function of the South-Korean block cipher standard LEA [21].

Unfortunately, cryptanalytic properties of S-functions (such as probabilities of differentials and correlations of linear approximations) are not always easy to compute. As a result, dealing with S-functions is often a burden for cryptanalysts. In addition, there is a risk that these difficulties artificially limit the design space, as designers may prefer suboptimal functions that are easier to analyze. These issues are exacerbated by a growing number of cryptanalytic techniques. To name just a few examples, apart from differential and linear properties, there is a need to compute the boomerang connectivity table [17], the differential-linear connectivity table [3], the quasidifferential transition matrix [10], the ultrametric integral transition matrix [13], ... of these functions.

Previous work. The literature on the analysis of specific S-functions such as modular addition, is extensive. The differential properties of addition modulo 2^n were first analyzed by Lipmaa and Moriai in 2001 [30]. Wallén later gave an efficient algorithm to compute correlations of linear approximations [43]. Much later, Schulte-Geers [41] reexplained these results using the CCZ equivalence of modular addition to a quadratic function. This approach was also used in [10] to

[7] The χ-function is an S-function after breaking its cyclic dependency on inputs.

compute the quasidifferential transition matrix of addition modulo 2^n. The differential properties of modular addition with a constant, which is *not* CCZ equivalent to a quadratic function, were analyzed in [1,2,33]. Braeken and Semaev [15] computed algebraic properties of modular addition, as well as modular addition with a constant. These results recently resurfaced [23] due to their significance for integral cryptanalysis. The differential-linear [38,39] and boomerang [26,38] connectivity tables of modular addition were only recently computed.

A systematic approach to analyze the cryptanalytic properties of S-functions is missing. Mouha, Velichkov, De Cannière and Preneel [37] showed how to compute the differential properties of arbitrary S-functions. In the setting of ARX constructions, Leurent [28] modeled the differential analysis of additions and XORs as S-functions and used finite-state machines to compute the probabilities of differential and boomerang characteristics and to detect incompatible ones. However, for other properties, significant gaps remain in the literature. A good example is the χ-function: although its linear and differential properties are well-understood because it is quadratic, this is not helpful to compute its boomerang connectivity table or its ultrametric integral transition matrix. Even the ordinary integral properties of the χ-function are not understood, although some related results such as the algebraic normal form of χ^{-1} are known [32].

In the last five years, the geometric approach to cryptanalysis [7, 8] has led to a uniform description of most or all cryptanalytic techniques. In particular, this framework provides a general definition of cryptanalytic properties [8, §2.3] that covers all examples mentioned above. From this point of view, there is some hope that there exists a 'universal formula' to compute properties of S-functions.

Contribution. This paper develops a systematic approach to evaluate properties of arbitrary S-functions (Mealy machines). In particular, our approach allows evaluating any cryptanalytic property in the sense of the geometric approach to cryptanalysis [8, §2.3], provided that the property is compatible with the way the input and output are split into chunks[8]. To the best of our knowledge, this covers all commonly-used properties, including for example the probabilities of differentials, d-differentials and boomerangs, the correlations of linear and differential-linear approximations, and integral or ultrametric integral properties.

More precisely, our main result (Theorem 2) states that for all such properties, there exist a column vector a, a row vector b, and matrices $M_1, \ldots, M_n$, such that the property evaluates to

$$b\, M_n \cdots M_2 M_1\, a\,. \tag{$\dagger$}$$

This is consistent with earlier isolated results for *specific* properties and *specific* S-functions such as the modular addition function. Note that ($\dagger$) leads to an algorithm to evaluate properties that has runtime linear in the input length.

Sections 2 to 4 develop the framework and prove our main theorem. Sections 5 to 8 then demonstrate it on concrete S-functions and applications.

[8] Formally, the property must be a pair of rank-one tensors (see Section 2.1 for details).

To prove this result, we start by slightly extending the geometric approach in Section 2 so that it can be used to analyze *partial functions*, i.e. functional relations that are only defined on a subset of their domain. The motivation for this is that, after fixing the input and output of an S-function, its state-update operation is still a partial function. This idea is developed in Section 3.

The point of view introduced in Sections 2 and 3 leads to a relatively clean proof of Theorem 2 and the relation (†). In addition, it provides a simple interpretation of the matrices M_i in terms of the partial state-update functions of the S-function. Apart from an optimal $\mathcal{O}(n)$-time algorithm to evaluate the property, (†) also leads to a sufficient condition for the existence of compact constraints for a family of properties (such as all linear approximations). Our condition covers earlier examples where this was known to be possible. It is based on simultaneous row- or column-monomialization of the matrices $M_1, \ldots, M_n$ by a change-of-basis. In the supplementary material, we provide tools to automate the computation of the matrices M_i, including an implementation of our algorithm to find a simplifying change-of-basis transformation if it exists.

In Section 5, we illustrate our results by analyzing several families of properties for well-known S-functions: modular addition, modular addition with a constant, the χ-function, the $\barwedge$-function [6], and the SHA-1 step function. Some of these examples are new results, including but not limited to the (i) ultrametric integral properties of modular addition, (ii) quasidifferential properties modular addition with a constant, and (iii) the integral properties of χ and $\barwedge$. Importantly, these are just examples; the broader point is that Theorem 2 provides a uniform formula for cryptographic properties of arbitrary S-functions.

Sections 6 to 8 contain more advanced applications of our result. Section 6 presents a boomerang distinguisher for the full Subterranean 2.0 permutation. This is a valid distinguisher for the permutation, but its data-complexity is too high to affect the security of any constructions based on it. Hence, the main goal of this application is to illustrate how Theorem 2 can be used to calculate the boomerang connectivity table of the χ function on 257 bits. As a side result, we also compute the correlations of quasi-3-differentials [44]. Section 7 shows how the correlations of conditional linear approximations used in partitioning-based differential-linear attacks on ARX-ciphers can be computed exactly rather than heuristically. This leads us to revisit the attacks on LEA from Asiacrypt 2023 [16]. Finally, in Section 8, we provide a new approach to derive the algebraic normal form of the inverse of the χ-function. This result was obtained earlier by Liu et al. [32], but with a technical proof. Our approach is quite general and relies only on the integral properties of the inverse of the χ-function, which we can compute using Theorem 2.

2 Geometric Approach for Partial Functions

The geometric approach to cryptanalysis was proposed by Beyne [7, 8] as a meta-approach to think about various cryptanalytic techniques. It has since been shown that this approach leads to a uniform description of common tech-

niques such as linear cryptanalysis [7], differential cryptanalysis [10], integral cryptanalysis [11,12], as well as many combined methods [24]. This is useful for our purposes, as it allows us to provide a general analysis of the properties of S-functions.

Sections 2.1 and 2.2 summarize a few concepts from the geometric approach that are used in this work. However, we slightly generalize some of them: whereas the geometric approach is primarily concerned with the properties of (total) functions, our analysis relies on the properties of *partial functions*. Section 2.3 shows that the results we need carry over to this setting.

2.1 Cryptanalytic properties

The geometric approach is essentially a dictionary between combinatorics and linear algebra. In particular, statements about functions $F\colon X \to Y$ between finite sets become statements about linear maps $T^F\colon \mathbb{K}[X] \to \mathbb{K}[Y]$ between vector spaces over a field $\mathbb{K}$. A formalization of this principle can be found in [9].

The free vector space $\mathbb{K}[X]$ on a finite set X consists of all linear combinations $\sum_{x\in X} u[x]\,\delta_x$, where the coordinates $u[x]$ are elements of $\mathbb{K}$ and δ_x are formal basis vectors. From the viewpoint of cryptanalysis, an element of $\mathbb{K}[X]$ represents an assignment of weights (elements of $\mathbb{K}$) to X. For a function $F\colon X \to Y$, the linear map $T^F\colon \mathbb{K}[X] \to \mathbb{K}[Y]$ is defined by $T^F\,\delta_x = \delta_{F(x)}$ for all x in X.

There is a dual point of view, based on the vector space $\mathbb{K}^X$ of $\mathbb{K}$-valued functions on a finite set X. It is the dual vector space of $\mathbb{K}[X]$, in the sense that every function $v\colon X \to \mathbb{K}$ uniquely extends to a linear function $\mathbb{K}[X] \to \mathbb{K}$ by mapping δ_x to $v(x)$. The vector space $\mathbb{K}^X$ has a standard basis $\{\delta^x \mid x \in X\}$, consisting of the functions δ^x defined by $\delta^x(t) = 1$ if $t = x$ and zero elsewhere. From the viewpoint of cryptanalysis, an element of $\mathbb{K}^X$ represents a way to probe or observe states in $\mathbb{K}[X]$. This interpretation leads to the following definition of cryptanalytic properties.

Definition 1 (Cryptanalytic property [8, Definition 2.6]). *A cryptanalytic property of a function $F\colon X \to Y$ is a pair (u, v) with u in $\mathbb{K}[X]$ and v in $\mathbb{K}^Y$. The evaluation of (u, v) is defined as $v(T^F u)$.*

From the point of view of Definition 1, the main result of this paper is to evaluate cryptanalytic properties when F is an S-function. For *arbitrary* u and v, this is not a reasonable goal for a normal function due to the size of X and Y. However, for S-functions, $X = X_1 \times \cdots \times X_n$ and $Y = Y_1 \times \cdots \times Y_n$. In this case, our results allow evaluating arbitrary properties of the form $u = u_1 \otimes \cdots \otimes u_n$ and $v = v_1 \otimes \cdots \otimes v_n$. Here, the tensor product $\otimes$ may be defined concretely by $u[(x_1, \ldots, x_n)] = u_1[x_1] \cdots u_n[x_n]$ and $v((x_1, \ldots, x_n)) = v_1(x_1) \cdots v_n(x_n)$.

2.2 Examples of cryptanalytic properties and bases

In practice, the vectors u and v in Definition 1 are often part of a basis for $\mathbb{K}[X]$ and $\mathbb{K}^Y$, respectively. Different choices of the basis lead to different techniques

such as linear or integral cryptanalysis. The following examples are used in the application sections of this paper. However, note that the results in Sections 3 and 4 are independent of the choice of basis.

Linear cryptanalysis. For linear cryptanalysis, one assumes X is a finite Abelian group. The basis for $\mathbb{K}^X$ consists of the $\mathbb{C}$-valued characters of X, and the basis for $\mathbb{K}[X]$ is its dual. When $X = \mathbb{F}_2^n$, we can work in $\mathbb{K} = \mathbb{R}$, thus the basis of characters for $\mathbb{R}^X$ is given by $\{\chi^u \mid u \in \mathbb{F}_2^n\}$, where

$$\chi^u(x) = (-1)^{u^\top x}.$$

The dual basis for $\mathbb{R}[X]$ is given by $\{\chi_u \mid u \in \mathbb{F}_2^n\}$, with $\chi_u[x] = (-1)^{u^\top x}/2^n$. Note that $\chi_u = \chi_{u_1} \otimes \cdots \otimes \chi_{u_n}$ and $\chi^u = \chi^{u_1} \otimes \cdots \otimes \chi^{u_n}$ for all u in $\mathbb{F}_2^n$.

Integral cryptanalysis. For ultrametric integral cryptanalysis, X is a ring isomorphic to a product of fields [13]. The field $\mathbb{K}$ is an extension of the p-adic numbers $\mathbb{Q}_p$, and the basis for $\mathbb{K}^X$ consists of the multiplicative characters of X. For $X = \mathbb{F}_2^n$, choosing $\mathbb{K} = \mathbb{Q}$ is sufficient and the basis for $\mathbb{Q}^X$ is $\{\mu^u \mid u \in \mathbb{F}_2^n\}$. Here,

$$\mu^u(x) = \tau(x^u),$$

with $\tau\colon \mathbb{F}_2 \to \mathbb{Q}$ the map given by $\tau(0) = 0$ and $\tau(1) = 1$. The notation x^u is defined by $x^u = x_1^{u_1} x_2^{u_2} \cdots x_n^{u_n}$. The dual basis $\{\mu_u \mid u \in \mathbb{F}_2^n\}$ is given by

$$\mu_u[x] = (-1)^{\mathrm{wt}(x+u)}\, \tau(u^x).$$

Note that the basis vectors satisfy $\mu_u = \mu_{u_1} \otimes \cdots \otimes \mu_{u_n}$ and $\mu^u = \mu^{u_1} \otimes \cdots \mu^{u_n}$. Ordinary integral cryptanalysis uses the same basis, but reduced modulo two. That is, $\mu^u(x) \equiv x^u \pmod 2$ and $\mu_u[x] \equiv u^x \pmod 2$.

Differential cryptanalysis. In the case of differential cryptanalysis, X is a set of pairs $G \times G$, where G is a finite Abelian group. This also means that a primitive F should be extended to a function on pairs $(x, y) \mapsto (F(x), F(y))$. The basis is known as the quasidifferential basis [8, 10]. For $X = \mathbb{F}_2^n \times \mathbb{F}_2^n$, it consists of the functions in $\mathbb{R}^X$

$$q^{(u,d)}((x, y)) = (-1)^{u^\top x}\, \delta_d(x + y).$$

The basis for $\mathbb{R}[X]$ consists of the vectors $q_{(u,d)}[(x, y)] = (-1)^{u^\top x}\, \delta_d(x + y)/2^n$.

2.3 Pushforward operator of a partial function

Recall that a partial function $F\colon X \nrightarrow Y$ is a function from a subset of X to Y. More precisely, for all x in X, the value of $F(x)$ is either an element of Y or *undefined*. The latter case will be denoted by $F(x) = \perp$. In other words, every partial function is equivalent to a function $X \to Y \cup \{\perp\}$.

Definition 2 (Pushforward operator). *The pushforward of a partial function $F\colon X \twoheadrightarrow Y$ is the linear map $T^F\colon \mathbb{K}[X] \to \mathbb{K}[Y]$ defined by*

$$T^F\,\delta_x = \begin{cases} \delta_{F(x)} & \text{if } F(x) \neq \perp\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

The matrix representation of T^F with respect to the standard bases of $\mathbb{K}[X]$ and $\mathbb{K}[Y]$ is called the *transition matrix* of F. Its rows and columns are indexed by elements of X and Y, respectively. Depending on the context, the notation T^F either refers to the pushforward operator of F or to its transition matrix.

The composition of partial functions F and G is again a partial function, with $F(G(x)) = \perp$ whenever $G(x) = \perp$. For ordinary functions, the pushforward is compatible with composition. This result extends to partial functions.

Theorem 1. *If $F = F_r \circ \cdots \circ F_2 \circ F_1$ is the composition of partial functions $F_1, F_2 \ldots, F_r$, then $T^F = T^{F_r} \cdots T^{F_2} T^{F_1}$.*

Proof. The proof is nearly identical to the proof in [8, Theorem 2.4]. It is sufficient to prove the result for $r = 2$. For all x in the domain of F,

$$T^{F_2} T^{F_1}\,\delta_x = \begin{cases} T^{F_2}\,\delta_{F_1(x)} & \text{if } F_1(x) \neq \perp\,, \\ 0 & \text{otherwise} \end{cases} = \begin{cases} \delta_{F_2(F_1(x))} & \text{if } F_2(F_1(x)) \neq \perp\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

By definition, the above is equal to $T^{F_2 \circ F_1}\,\delta_x$. □

Theorem 1 is independent of the choice of basis. In particular, if $\mathscr{B}_1, \ldots, \mathscr{B}_{r+1}$ are change-of-basis transformations and $B^{F_i} = \mathscr{B}_{i+1} T^{F_i} \mathscr{B}_i^{-1}$, then the map $B^F = \mathscr{B}_{r+1} T^F \mathscr{B}_1^{-1}$ satisfies $B^F = B^{F_r} \cdots B^{F_2} B^{F_1}$.

3 S-Functions

This section establishes a few properties of S-functions that play an important role in Section 4. An S-function is a function $X_1 \times \cdots \times X_n \to Y_1 \times \cdots \times X_n$ such that the i^{th} coordinate of the output only depends on the i^{th} coordinate of the input and on a constant-size state that is updated iteratively based on the current input and the previous state. A circuit representation of S-functions is shown in Figure 1 of Section 1. The terminology *state function* or *S-function* was introduced by Mouha, Velichkov, De Cannière and Preneel [37].

It will be useful to consider the initial state as an additional input and the final state as an additional output. If this convention is adopted, an S-function is a function $S \times X_1 \times \cdots \times X_n \to T \times Y_1 \times \cdots \times Y_n$. Here, S is the set of initial states, and T is the set of final states.

Example 1 (Modular addition). Addition modulo 2^n is an S-function with $S = T = Y_i = \mathbb{F}_2$ and $X_i = \mathbb{F}_2^2$. The state corresponds to the carry, and all of the functions F_i are identical. This description of modular addition allows for a nonzero input carry bit.

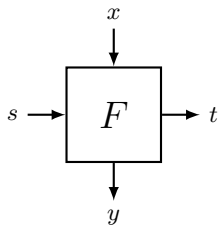


Fig. 2: The restriction of $F\colon S \times X \to T \times Y$ to fixed x and y yields a parameterized partial function $F(\,\cdot\,;\, x,\, y)\colon S \to T$ with parameters x and y.

3.1 S-functions as partial functions with parameters

As discussed above, an S-function is a function of the form $F\colon S \times X \to T \times Y$ with two inputs and two outputs (see Figure 2). Every such function F can alternatively be understood as an indexed family of partial functions $F(\,\cdot\,;\, x,\, y)\colon S \rightharpoonup T$ with parameters x and y. In more detail, $F(\,\cdot\,;\, x,\, y)$ is defined as follows:

$$F(s; x, y) = \begin{cases} t & \text{if } F(s, x) = (t, y) \text{ for some } t \text{ in } T\,, \\ \bot & \text{otherwise}\,. \end{cases}$$

Not every family of partial functions $F(\,\cdot\,;\, x,\, y)\colon S \rightharpoonup T$ gives rise to a function $F\colon S \times X \to T \times Y$. This requires that, for every (s, x) in $S \times X$, there exists a unique (t, y) in $T \times Y$ so that $t = F(s; x, y)$. This property is preserved under composition of partial functions. Hence, Definition 3 below is a valid alternative definition of S-functions.

Definition 3 (S-function). *Let $S_1, \ldots, S_{n+1}$, $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_n$ be finite sets with $X = \prod_{i=1}^{n} X_i$ and $Y = \prod_{i=1}^{n} Y_i$. For all functions $F_1, \ldots, F_n$ with $F_i\colon S_i \times X_i \to S_{i+1} \times Y_i$, the function $F\colon S_1 \times X \to S_{n+1} \times Y$ defined by $F(\,\cdot\,;\, x,\, y) = F_n(\,\cdot\,;\, x_n,\, y_n) \circ \cdots \circ F_1(\,\cdot\,;\, x_1,\, y_1)$ for all x in X and y in Y, is called an S-function. This is denoted by $F = \text{S-}\textsc{function}[F_1, \ldots, F_n]$.*

As shown in Section 3.2 below, Definition 3 is a useful characterization of S-functions to analyze their properties. This is because it allows us to view S-functions as compositions of (parameterized) partial functions, making it possible to apply Theorem 1.

Example 2 (Modular addition). Like in Example 1, consider the S-function that adds two bitvectors modulo 2^n. The truth table of the functions $F_1 = F_2 = \cdots = F_n$ is given in Table 1. The parametrized partial functions are defined for only one input. For example, $F(\,\cdot\,;\, \mathtt{00},\, \mathtt{0})$ is given by $0 \mapsto 0$ and $1 \mapsto \bot$.

Remark 1 (S-functions with a cycle). Some functions have a cyclic dependence on their inputs: they are essentially S-functions in which the initial state is equal to the final state. One example is the χ-function on $\mathbb{F}_2^n$. It is defined by $\chi\colon x \mapsto y$

Table 1: The truth table of $F_i\colon (s_i, x_i) \mapsto (s_{i+1}, y_i)$ for modular addition.

(s_i, x_i)	0 00	1 00	0 10	1 10	0 01	1 01	0 11	1 11
(s_{i+1}, y_i)	0 0	0 1	0 1	1 0	0 1	1 0	1 0	1 1

with $y_i = x_i + (x_{i-1}+1)x_{i-2}$, where $x_0 = x_n$ and $x_{-1} = x_{n-1}$. Section 4.4 shows that the properties of functions of this type can be computed from the properties of a standard S-function by breaking the cyclic dependence on the inputs.

3.2 Pushforward operator of an S-function

By Definition 3, every S-function $F = \text{S-FUNCTION}[F_1, \ldots, F_n]$ satisfies

$$F(\,\cdot\,;\, x,\, y) = F_n(\,\cdot\,;\, x_n,\, y_n) \circ \cdots \circ F_2(\,\cdot\,;\, x_2,\, y_2) \circ F_1(\,\cdot\,;\, x_1,\, y_1)\,.$$

Applying Theorem 1 to this equality yields the following lemma.

Lemma 1. *Let $S_1, \ldots, S_{n+1}$, $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_n$ be finite sets and $F_i\colon S_i \times X_i \to S_{i+1} \times Y_i$ functions. If $F = S\text{-FUNCTION}[F_1, \ldots, F_n]$, then*

$$T^{F(\,\cdot\,;\,x,\,y)} = T^{F_n(\,\cdot\,;\,x_n,\,y_n)} \cdots T^{F_2(\,\cdot\,;\,x_2,\,y_2)} T^{F_1(\,\cdot\,;\,x_1,\,y_1)}\,,$$

for all $x = (x_1, \ldots, x_n)$ in $X_1 \times \cdots \times X_n$ and $y = (y_1, \ldots, y_n)$ in $Y_1 \times \cdots \times Y_n$.

In Section 4, a generalization of Lemma 1 (that immediately follows from it) will be used to evaluate arbitrary properties of S-functions by multiplying a sequence of matrices.

4 Cryptanalytic Properties of S-Functions

This section contains our main result, which allows for the evaluation of the cryptanalytic properties of S-functions. Recall that a cryptanalytic property for a function F is a pair (u, v), with u as an element of the free $\mathbb{K}$-vector space on the domain of F and v as a $\mathbb{K}$-valued function on the codomain of F. The results in this section allow us to evaluate any property $(a \otimes u, b \otimes v)$ of an S-function $F\colon S_1 \times X_1 \times \cdots \times X_n \to S_{n+1} \times Y_1 \times \cdots \times Y_n$, as long as $u = u_1 \otimes \cdots \otimes u_n$ and $v = v_1 \otimes \cdots \otimes v_n$.

4.1 Generalized pushforward for partial functions with parameters

Let $F\colon S \times X \to T \times Y$ be a function. As discussed in Section 3, F gives rise to an indexed family of partial functions $F(\,\cdot\,;\, x,\, y)\colon S \to T$. The following definition generalizes the pushforward operator of $F(\,\cdot\,;\, x,\, y)$. Specifically, for all u in $\mathbb{K}[X]$ and v in $\mathbb{K}^Y$, let

$$T^F(\,\cdot\,;\, u,\, v) = \sum_{x \in X} \sum_{y \in Y} u[x]\, v(y)\, T^{F(\,\cdot\,;\,x,\,y)}\,.$$

In particular, $T^F(\,\cdot\,;\,\delta_x,\,\delta^y) = T^{F(\,\cdot\,;\,x,\,y)}$. The evaluation of $T^F(\,\cdot\,;\,u,\,v)$ at a vector a in $\mathbb{K}[S]$ will be denoted by $T^F(a\,;\,u,v)$, or equivalently by $T^F(\,\cdot\,;\,u,\,v)a$ using matrix-vector product notation.

Example 3 (Modular addition). Let F_i be the state-update function for addition modulo 2^n as in Examples 1 and 2. Based on the truth-table from Example 2, the transition matrices of the partial functions $F_i(\,\cdot\,;\,x,\,y)$ are as follows:

$$
\begin{aligned}
T^{F_i}(\,\cdot\,;\,\delta_{00},\,\delta^0) &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, & T^{F_i}(\,\cdot\,;\,\delta_{00},\,\delta^1) &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \\
T^{F_i}(\,\cdot\,;\,\delta_{01},\,\delta^0) &= \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, & T^{F_i}(\,\cdot\,;\,\delta_{01},\,\delta^1) &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \\
T^{F_i}(\,\cdot\,;\,\delta_{10},\,\delta^0) &= \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, & T^{F_i}(\,\cdot\,;\,\delta_{10},\,\delta^1) &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \\
T^{F_i}(\,\cdot\,;\,\delta_{11},\,\delta^0) &= \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, & T^{F_i}(\,\cdot\,;\,\delta_{11},\,\delta^1) &= \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.
\end{aligned}
$$

To analyze linear approximations for the modular addition function, we consider the matrices $T^{F_i}(\,\cdot\,;\,\chi_u,\,\chi^v)$, where u and v are masks. For example,

$$
\begin{aligned}
T^{F_i}(\,\cdot\,;\,\chi_{01},\,\chi^0) = &\frac{1}{4}\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} - \frac{1}{4}\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} + \frac{1}{4}\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} - \frac{1}{4}\begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \\
&+\frac{1}{4}\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} - \frac{1}{4}\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + \frac{1}{4}\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} - \frac{1}{4}\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.
\end{aligned}
$$

For completeness, we provide the matrices for all masks u and v below:

$$
\begin{aligned}
T^{F_i}(\,\cdot\,;\,\chi_{00},\,\chi^0) &= \frac{1}{4}\begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix}, & T^{F_i}(\,\cdot\,;\,\chi_{00},\,\chi^1) &= \frac{1}{4}\begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix}, \\
T^{F_i}(\,\cdot\,;\,\chi_{01},\,\chi^0) &= \frac{1}{4}\begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}, & T^{F_i}(\,\cdot\,;\,\chi_{01},\,\chi^1) &= \frac{1}{4}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \\
T^{F_i}(\,\cdot\,;\,\chi_{10},\,\chi^0) &= \frac{1}{4}\begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}, & T^{F_i}(\,\cdot\,;\,\chi_{10},\,\chi^1) &= \frac{1}{4}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \\
T^{F_i}(\,\cdot\,;\,\chi_{11},\,\chi^0) &= \frac{1}{4}\begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}, & T^{F_i}(\,\cdot\,;\,\chi_{11},\,\chi^1) &= \frac{1}{4}\begin{bmatrix} 3 & -1 \\ 1 & -3 \end{bmatrix}.
\end{aligned}
$$

The definition of $T^F(\,\cdot\,;\,u,\,v)$ is motivated by the following lemma, which shows that the linear map $T^F(\,\cdot\,;\,u,\,v)$ contains enough information to evaluate all properties of F that are of the form $(a \otimes u, b \otimes v)$.

Lemma 2. *Let $F\colon S \times X \to T \times Y$ be a function. For all $a \otimes u$ in $\mathbb{K}[S \times X]$ and $b \otimes v$ in $\mathbb{K}[T \times Y]$, it holds that*

$$
(b \otimes v)\Big(T^F(a \otimes u)\Big) = b\Big(T^F(a\,;\,u,v)\Big).
$$

Proof. By the definition of $F(\,\cdot\,;\,x,\,y)$ and $T^F(\,\cdot\,;\,\delta_x,\,\delta^y) = T^{F(\,\cdot\,;\,x,\,y)}$, we have

$$\big(\delta^t \otimes \delta^y\big)\Big(T^F(\delta_s \otimes \delta_x)\Big) = \delta^t\Big(T^F(\delta_s\,;\,\delta_x, \delta^y)\Big)\,.$$

The result now follows from multilinearity. In particular, for the left-hand side,

$$(b \otimes v)\Big(T^F(a \otimes u)\Big) = \sum_{s\in S}\sum_{x\in X} a[s]u[x] \sum_{t\in T}\sum_{y\in Y} b(t)v(y)\,\big(\delta^t \otimes \delta^y\big)\Big(T^F(\delta_s \otimes \delta_x)\Big)\,.$$

Similarly, for the right-hand side,

$$b\Big(T^F(a\,;\,u, v)\Big) = \sum_{s\in S}\sum_{x\in X} a[s]u[x] \sum_{t\in T}\sum_{y\in Y} b(t)v(y)\,\delta^t\Big(T^F(\delta_s\,;\,\delta_x, \delta^y)\Big)\,.$$

Hence, the equality follows. □

4.2 Main result

Lemma 2 makes it possible to evaluate properties of F in terms of those of the parameterized partial functions $F(\,\cdot\,;\,x,\,y)$. Together with the following generalization of Lemma 1, this leads to the main result of this section.

Lemma 3. *Let $F_1, \ldots, F_n$ be functions $F_i\colon S_i \times X_i \to S_{i+1} \times Y_i$. The function $F = S\text{-}\textsc{function}[F_1, \ldots, F_n]$ satisfies*

$$T^F(\,\cdot\,;\,u,\,v) = T^{F_n}(\,\cdot\,;\,u_n,\,v_n)\cdots T^{F_2}(\,\cdot\,;\,u_2,\,v_2)T^{F_1}(\,\cdot\,;\,u_1,\,v_1)\,,$$

for all $u = u_1 \otimes \cdots \otimes u_n$ in $\mathbb{K}[X_1 \times \cdots \times X_n]$ and $v = v_1 \otimes \cdots \otimes v_n$ in $\mathbb{K}^{Y_1 \times \cdots \times Y_n}$.

Proof. By Lemma 1, it holds that

$$T^F(\,\cdot\,;\,\delta_x,\,\delta^y) = T^{F_n}(\,\cdot\,;\,\delta_{x_n},\,\delta^{y_n})\cdots T^{F_2}(\,\cdot\,;\,\delta_{x_2},\,\delta^{y_2})T^{F_1}(\,\cdot\,;\,\delta_{x_1},\,\delta^{y_1})\,.$$

Plugging this into the definition of $T^F(\,\cdot\,;\,u,\,v)$ yields the following equality:

$$T^F(\,\cdot\,;\,u,\,v) = \sum_{x\in X}\sum_{y\in Y} u[x]v(y)\,T^{F_n}(\,\cdot\,;\,\delta_{x_n},\,\delta^{y_n})\cdots T^{F_1}(\,\cdot\,;\,\delta_{x_1},\,\delta^{y_1})$$

The right-hand side of the equation above equals

$$\underbrace{\sum_{\substack{x_n\in X_n\\ y_n\in Y_n}} u_n[x_n]v_n(y_n)\,T^{F_n}(\,\cdot\,;\,\delta_{x_n},\,\delta^{y_n})}_{T^{F_n}(\,\cdot\,;\,u_n,\,v_n)} \cdots \underbrace{\sum_{\substack{x_1\in X_1\\ y_1\in Y_1}} u_1[x_1]v_1(y_1)\,T^{F_1}(\,\cdot\,;\,\delta_{x_1},\,\delta^{y_1})}_{T^{F_1}(\,\cdot\,;\,u_1,\,v_1)}\,.$$

This is the desired result. □

Lemma 3 leads to the following theorem, which is the main result of this section. It provides a formula for the evaluation of any cryptanalytic property $(a \otimes u_1 \otimes \cdots \otimes u_n, b \otimes v_1 \otimes \cdots \otimes v_n)$ of an S-function.

Theorem 2. *Let $F_1, \ldots, F_n$ be functions $F_i \colon S_i \times X_i \to S_{i+1} \times Y_i$. The function $F = S\text{-}\textsc{function}[F_1, \ldots, F_n]$ satisfies*

$$(b \otimes v)\Big(T^F\,(a \otimes u)\Big) = b\Big(T^{F_n}(\,\cdot\,;\,u_n,\,v_n)\cdots T^{F_2}(\,\cdot\,;\,u_2,\,v_2)T^{F_1}(\,\cdot\,;\,u_1,\,v_1)\,a\Big)\,,$$

for all a in $\mathbb{K}[S_1]$, u_i in $\mathbb{K}[X_i]$, b in $\mathbb{K}^{S_{n+1}}$ and v_i in $\mathbb{K}^{Y_i}$.

Proof. The result follows by combining Lemmas 2 and 3. □

4.3 Comparison with previous work

Theorem 2 expresses the evaluation of cryptographic properties of S-functions as a product of a column vector, n matrices, and finally a row vector. For the differential properties of S-functions, a similar expression was obtained by Mouha et al. [37]. In an earlier paper, Lipmaa, Wallén and Dumas [31] obtained a similar expression for the additive differential probability of exclusive-or. In the context of ARX constructions, Leurent [28] used a closely related finite-state-machine analysis of S-functions, but only for differential and boomerang characteristics, where the matrices again admit a path-counting interpretation.

A major difference between our approach and earlier work such as [31,37] is that for arbitrary properties, the matrices $T^{F_i}(\,\cdot\,;\,u_i,\,v_i)$ in Theorem 2 are not necessarily column-stochastic. Indeed, in [31,37], the matrices originate from a probabilistic interpretation of the problem or (what is equivalent, after scaling) as the adjacency matrix of a graph in which paths must be counted. However, for many important cryptanalytic properties, such an interpretation is not available.

An interesting exception is the calculation of the correlations of linear approximations for the modular addition function by Wallén [43]. This result was later reinterpreted by Nyberg and Wallén [40, Appendix A] in terms of a product of (non-stochastic) matrices. However, in this case, the interpretation of how the matrices arise in the proof is less direct.

4.4 Boundary conditions and S-functions with a cycle

Theorem 2 allows evaluating arbitrary properties not only with respect to the input and output of the S-function, but also its initial and final state. In many cases, the initial state is simply a fixed value s. In this case, one can take $a = \delta_s$. Likewise, the final state is often not observed, so that $b = \sum_{s \in S_{n+1}} \delta^s$.

Example 4 (Modular addition). To compute the correlation of a linear approximation for modular addition, one should choose a and b as follows:

$$a = \delta_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad b = \chi^0 = \delta^0 + \delta^1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

This corresponds to fixing the first carry bit to zero, and to a linear approximation with a zero mask on the final carry bit.

Algorithm 1 Evaluate a cryptographic property $(a \otimes u, b \otimes v)$ of an S-function $F = \text{S-FUNCTION}[F_1, \ldots, F_n]$. Assumes $u = u_1 \otimes \cdots \otimes u_n$ and $v = v_1 \otimes \cdots \otimes v_n$.

▷ *Compute matrix representations M_i of the maps $T^{F_i}(\,\cdot\,; u_i, v_i)$.* ◁
Allocate matrices $M_1, \ldots, M_n$ with coordinates indexed by elements of $S_{i+1} \times S_i$
for $i = 1, 2, \ldots, n$ **do**
    ▷ *Compute the matrix M_i* ◁
    for $(s_i, x_i) \in S_i \times X_i$ **do**
        $(s_{i+1}, y_i) \leftarrow F_i(s_i, x_i)$
        $M_i[s_{i+1}, s_i] \leftarrow M_i[s_{i+1}, s_i] + u_i[x_i] v_i(y_i)$
▷ *Compute matrix-vector products to evaluate the property.* ◁
$x \leftarrow a$
for $i = 1, 2, \ldots, n$ **do**
    $x \leftarrow M_i \times x$
return $b(x)$ ▷ *In practice, b may be stored as a row vector, so the result is $b \times x$.*

However, there are S-functions with 'boundary conditions' that lead to more interesting choices of a and b. In particular, the choice of a and b allows us to apply the approach of Sections 3 and 4 to S-functions with a cycle such as the χ-function (see Remark 1). In the typical case, a cycle implies that the input and output state are identical, as the final state is fed back to the initial state. Hence, we must impose the condition $a = \delta_s$ and $b = \delta^s$. However, since the value of s is free, we should apply Theorem 2 to each pair $(a, b) = (\delta_s, \delta^s)$ and sum the results over all s in $S_1 = S_{n+1}$. Additional details are given in Section 5.3, where we analyze the properties of the χ-function.

4.5 Efficient evaluation of properties

Let $F = \text{S-FUNCTION}[F_1, \ldots, F_n]$ be an S-function with $F_i \colon S_i \times X_i \to S_{i+1} \times Y_i$. Assume that $|X_i|$, $|Y_i|$ and $|S_i|$ are constant. Theorem 2 shows that properties of F can be evaluated using n constant-size matrix-vector products. Hence, this approach yields an $\mathcal{O}(n)$-time algorithm to evaluate a given property. This is optimal from the point of view of bit-operations, despite the fact that there may exist algorithms that require a smaller number of *word operations*. A detailed description is given in Algorithm 1.

Although Algorithm 1 provides an $\mathcal{O}(n)$-time method to evaluate cryptanalytic properties of S-functions, this is often not sufficient to efficiently search for properties. Indeed, this often requires a compact formula for the evaluation of properties that can be converted to a SAT or MILP model. Section 4.6 analyzes the circumstances under which this is possible.

4.6 Compact constraints for properties

In some cases, such as for the linear and quasidifferential properties of modular addition, there exist more compact formulas than Theorem 2. These formulas are particularly suitable for use in constraint programming models.

Let $F_i \colon S_i \times X_i \to S_{i+1} \times Y_i$ be the state-update functions of an S-function $F = \text{S-FUNCTION}[F_1, \dots, F_n]$. Assume that $S_1 = S_2 = \cdots = S_n = S$, like for all examples in Section 5. Theorem 2 is independent of the choice of basis. In particular, let $\mathscr{B}$ be an arbitrary invertible linear transformation on $\mathbb{K}[S_i]$. If $B^{F_i}(\,\cdot\,;\, u_i,\, v_i) = \mathscr{B}\, T^{F_i}(\,\cdot\,;\, u_i,\, v_i)\, \mathscr{B}^{-1}$, then Lemma 3 becomes

$$B^F(\,\cdot\,;\, u,\, v) = B^{F_n}(\,\cdot\,;\, u_n,\, v_n) \cdots B^{F_2}(\,\cdot\,;\, u_2,\, v_2) B^{F_1}(\,\cdot\,;\, u_1,\, v_1)\,.$$

Theorem 2 can be modified accordingly, with a and b replaced by $\mathscr{B}\, a$ and $\mathscr{B}^{-t}\, b$ respectively. As the following example shows, applying a change of basis can considerably simplify the above product of matrices.

Example 5 (Modular addition). For the state-update function $F_i \colon \mathbb{F}_2 \times \mathbb{F}_2^2 \to \mathbb{F}_2 \times \mathbb{F}_2^2$ of addition modulo 2^n, the Fourier transformation simultaneously monomializes all of the matrices $T^{F_i}(\,\cdot\,;\, \chi_u,\, \chi^v)$. A *monomial matrix* is a product of a diagonal matrix and a permutation matrix. For example,

$$B^{F_i}(\,\cdot\,;\, \chi_{11},\, \chi^1) = \underbrace{\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}}_{\mathscr{F}} \underbrace{\frac{1}{4}\begin{bmatrix} 3 & -1 \\ 1 & -3 \end{bmatrix}}_{T^{F_i}(\,\cdot\,;\, \chi_{11},\, \chi^1)} \underbrace{\frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}}_{\mathscr{F}^{-1}} = \begin{bmatrix} 0 & 1 \\ \frac{1}{2} & 0 \end{bmatrix}.$$

Similarly, for the other matrices,

$$B^{F_i}(\,\cdot\,;\, \chi_{00},\, \chi^0) = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{2} \end{bmatrix}, \qquad B^{F_i}(\,\cdot\,;\, \chi_{00},\, \chi^1) = \begin{bmatrix} 0 & 0 \\ -\frac{1}{2} & 0 \end{bmatrix},$$

$$B^{F_i}(\,\cdot\,;\, \chi_{01},\, \chi^0) = \begin{bmatrix} 0 & 0 \\ \frac{1}{2} & 0 \end{bmatrix}, \qquad B^{F_i}(\,\cdot\,;\, \chi_{01},\, \chi^1) = \begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{2} \end{bmatrix},$$

$$B^{F_i}(\,\cdot\,;\, \chi_{10},\, \chi^0) = \begin{bmatrix} 0 & 0 \\ \frac{1}{2} & 0 \end{bmatrix}, \qquad B^{F_i}(\,\cdot\,;\, \chi_{10},\, \chi^1) = \begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{2} \end{bmatrix},$$

$$B^{F_i}(\,\cdot\,;\, \chi_{11},\, \chi^0) = \begin{bmatrix} 0 & 0 \\ 0 & -\frac{1}{2} \end{bmatrix}, \qquad B^{F_i}(\,\cdot\,;\, \chi_{11},\, \chi^1) = \begin{bmatrix} 0 & 1 \\ \frac{1}{2} & 0 \end{bmatrix}.$$

All of the above matrices are monomial matrices. In particular, every column and row contains at most one nonzero coordinate. Hence, the matrix products reduce to scalar products.

In Example 5, the change of basis $\mathscr{B}$ simultaneously monomializes all of the matrices. Assuming the product is evaluated from right to left, the same simplification is achieved if every column contains at most one nonzero coordinate. For convenience, we call a matrix of this form *column-monomial*. Evaluating the product from right to left similarly leads to a notion of row-monomial matrices. Section A contains an efficient algorithm to find a basis that simultaneously column-monomializes a set of matrices or to show that this is impossible.

If the matrices $T^{F_i}(\,\cdot\,;\, u,\, v)$ are simultaneously column-monomializable as in Example 5, this reduces the cost of Algorithm 1 by a constant factor. This is useful, but it is not the reason that column-monomial matrices lead to efficient constraint programming formulas. By Algorithm 1, the condition that the

evaluation of a property takes a particular value is a formula of size $\mathcal{O}(n)$. However, the number of possible values to which a property can evaluate is equally important.

Suppose that the number of *distinct* functions $F_1, F_2 \ldots, F_n$ is $\mathcal{O}(1)$ as $n \to \infty$. If the matrices $T^{F_i}(\,\cdot\,; u, v)$ are simultaneously column-monomializable, then every matrix multiplication in Theorem 2 amounts to a scalar product. The number of distinct scalars involved in the product is $\mathcal{O}(1)$, so the overall number of products that can be obtained by multiplying these scalars is at most

$$\binom{\mathcal{O}(1) + n - 1}{n} = \mathcal{O}(\mathrm{poly}(n))\,.$$

In particular, the overall size of the constraints required to model the evaluation of the property is $\mathcal{O}(n\mathrm{poly}(n)) = \mathcal{O}(\mathrm{poly}(n))$.

5 Application to Several Popular S-Functions

In this section, we apply the results from Section 4 to several concrete, widely-used S-functions. The overall approach is to use the first part of Algorithm 1 to compute the matrix-representation of $T^{F_i}(\,\cdot\,; u_i, v_i)$ for all u_i and v_i from a particular basis. Theorem 2 then provides a formula for the evaluation of the corresponding cryptanalytic properties.

As proof of concept, we analyze the properties of the S-functions in this sections with respect to three representative cryptanalytic techniques: linear, differential, and (ultrametric) integral cryptanalysis. These correspond to the bases listed in Section 2.1. Other properties such as differential-linear approximations, boomerangs and d-differentials [44] can be handled in the same way.

5.1 Modular addition

Addition modulo 2^n is the most widely studied S-functions, with an established line of work studying its properties. The case of linear cryptanalysis was already discussed in Example 3, and this result was first obtained by Wallén [43]. Lipmaa and Moriai first computed the probabilities of differentials [30], and the correlations of quasidifferentials were computed in [10] using the CCZ-equivalence of modular addition to a quadratic function. The integral properties of modular addition were described in [23], based on results of Braeken and Semaev [15]. For ultrametric integral cryptanalysis, the result in this section is new.

The modular addition is illustrated in Figure 3. The state-update functions $F_i \colon \mathbb{F}_2 \times \mathbb{F}_2^2 \to \mathbb{F}_2 \times \mathbb{F}_2$ are all the same, and their truth table was already given in Example 2 (Table 1). Alternatively, the function $F_i \colon (s_i, x_i \| y_i) \mapsto (s_{i+1}, z_i)$ is given by its algebraic normal form,

$$\begin{aligned} s_{i+1} &= x_i y_i + x_i s_{i-1} + y_i s_{i-1}\,, \\ z_i &= x_i + y_i + s_i\,. \end{aligned}$$

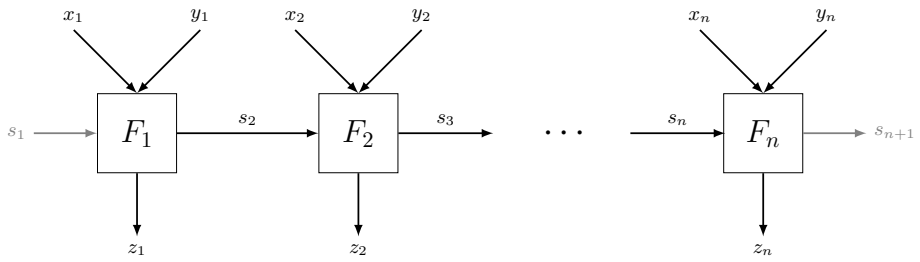


Fig. 3: Modular addition as an S-function. The state-update function takes input (x_i, y_i, s_{i-1}) and outputs (z_i, s_i). The states $s_1, \ldots, s_n$ are the carry bits.

Linear cryptanalysis. The initial state of the modular addition must be zero, but no mask is placed on the final state. Hence, to compute the correlations of linear approximations, the initial state is given by $a = \delta_0$. The final 'probe function' b is given by $b = \chi^0 = \delta^0 + \delta^1$. Given an input mask $u = (u_1, \ldots, u_n)$ with u_i in $\mathbb{F}_2^2$ and an output mask $v = (v_1, v_2, \ldots, v_n)$ with v_i in $\mathbb{F}_2$, Theorem 2 shows that the correlation is equal to

$$\begin{bmatrix}1\ 1\end{bmatrix}\ T^{F_n}(\,\cdot\,;\,\chi_{u_n},\,\chi^{v_n})\cdots T^{F_2}(\,\cdot\,;\,\chi_{u_2},\,\chi^{v_2})T^{F_1}(\,\cdot\,;\,\chi_{u_1},\,\chi^{v_1})\ \begin{bmatrix}1\\0\end{bmatrix}$$

The eight matrices $T^{F_i}(\,\cdot\,;\,\chi_{u_i},\,\chi^{v_i})$ were already listed in Example 3. As shown in Example 5, these matrices are simultaneously monomializable. This implies the existence of compact constraints.

Differential cryptanalysis. For differential cryptanalysis, we need to analyze the modular addition function on pairs. That is, the state-update functions are given by $G_i\colon \mathbb{F}_2^2 \times \mathbb{F}_2^4 \to \mathbb{F}_2^2 \times \mathbb{F}_2^4$, with

$$G_i(s_i\|s_i',\,x_i\|y_i\|x_i'\|y_i') = F_i(s_i,\,x_i\|y_i)\|F_i(s_i',\,x_i'\|y_i')\,.$$

The initial state is zero, so $a = \delta_{00}$. The final state is not observed, i.e. $b = \delta^{00} + \delta^{01} + \delta^{10} + \delta^{11}$. Hence, for an input mask-difference pair (u, d) in $\mathbb{F}_2^{2n} \times \mathbb{F}_2^{4n}$ and an output mask-difference pair (v, e) in $\mathbb{F}_2^n \times \mathbb{F}_2^n$, the correlation of the corresponding quasidifferential is

$$\begin{bmatrix}1\ 1\ 1\ 1\end{bmatrix} T^{G_n}(\,\cdot\,;\,q_{u_n,d_n},\,q^{v_n,e_n})\cdots T^{G_1}(\,\cdot\,;\,q_{u_1,d_1},\,q^{v_1,e_1})\begin{bmatrix}1\ 0\ 0\ 0\end{bmatrix}^t\,.$$

In a total we have 64 matrices $G^{F_i}(\,\cdot\,;\,q_{u_i,d_i},\,q^{v_i,e_i})$ since $u, d \in \mathbb{F}_2^2$ and $v, e \in \mathbb{F}_2$. We list these matrices in Appendix D. Among the 64 matrices, those with $v_i = u_i = 0$ are the matrices for ordinary differentials. They correspond to the matrices obtained by Mouha et al. in [37].

Using the algorithm in Section A, we find that all 64 matrices can be simultaneously row-monomialized. As a result, we obtain compact constraints for the quasidifferential transition matrix of modular addition. This corresponds to the result obtained in [10] using CCZ-equivalence.

Ultrametric integral cryptanalysis. The ultrametric integral properties of modular addition are similar to the linear properties, except that we use the ultrametric integral basis rather than the Fourier basis. For an input exponent $u = (u_1, u_2, \ldots, u_n)$ with u_i in $\mathbb{F}_2^2$ and an output exponent $v = (v_1, v_2, \ldots, v_n)$ with v_i in $\mathbb{F}_2$, the correlation is equal to

$$\begin{bmatrix}1 & 1\end{bmatrix} \; T^{F_n}(\,\cdot\,;\,\mu_{u_n},\,\mu^{v_n}) \cdots T^{F_2}(\,\cdot\,;\,\mu_{u_2},\,\mu^{v_2}) T^{F_1}(\,\cdot\,;\,\mu_{u_1},\,\mu^{v_1}) \; \begin{bmatrix}1\\0\end{bmatrix}$$

The eight matrices $T^{F_i}(\,\cdot\,;\,\mu_{u_i},\,\mu^{v_i})$ are as follows:

$$T^{F_i}(\,\cdot\,;\,\mu_{00},\,\mu^{0}) = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\mu_{00},\,\mu^{1}) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix},$$

$$T^{F_i}(\,\cdot\,;\,\mu_{01},\,\mu^{0}) = \begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\mu_{01},\,\mu^{1}) = \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix},$$

$$T^{F_i}(\,\cdot\,;\,\mu_{10},\,\mu^{0}) = \begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\mu_{10},\,\mu^{1}) = \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix},$$

$$T^{F_i}(\,\cdot\,;\,\mu_{11},\,\mu^{0}) = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\mu_{11},\,\mu^{1}) = \begin{bmatrix} -2 & 1 \\ 0 & 1 \end{bmatrix}.$$

Taking all elements of these matrices modulo two, we obtain the matrices used to evaluate ordinary integral properties over $\mathbb{F}_2$. The above matrices can be simultaneously column-monomialized modulo two and modulo four, again yielding compact constraints. For the modulo two case, the constraints correspond to those given in [23]. However, the algorithm from Section A shows that simultaneous row or column monomialization is not possible over any field extension of the rational numbers.

5.2 Modular addition with a constant

Let c be a constant in $\mathbb{F}_2^n$. The function $x \mapsto x \boxplus c$, where $\boxplus$ is the modular addition of bitvectors that was discussed in Section 5.1, is an S-function. Although addition with a constant is closely related to the modular addition function, its properties are often different and depend on the constant c.

The differential and linear properties of modular addition with a constant were first analyzed by Miyano [35] without an explicit algorithm, and bitvector constraints[9] for the differential case were given by Azimi et al. at Asiacrypt 2020 [1]. The results we provide below for linear, quasidifferential and ultrametric integral properties are new. For ordinary integral cryptanalysis, the same formula can be derived from results of Braeken and Semaev [15].

The main technical difference with Section 5.1 is that the state-update functions $F_i \colon \mathbb{F}_2 \times \mathbb{F}_2 \to \mathbb{F}_2 \times \mathbb{F}_2$ are not all the same, because they depend on the bits c_i. In particular, F_i is given by $F_i(s_i, x_i) = (s_{i+1}, y_i)$, where

$$y_i = x_i + s_i + c_i \,,$$

[9] The constraints model an approximation of the differential probabilities.

$$s_{i+1} = x_i c_i + x_i s_i + c_i s_i \,.$$

This expression is sufficient to compute the matrices $T^{F_i}(\,\cdot\,;\,u,\,v)$ in Theorem 2.

Linear cryptanalysis. Like in Section 5.1, the correlation of a linear approximation (u, v) with u in $\mathbb{F}_2^n$ and v in $\mathbb{F}_2^n$ is given by

$$\begin{bmatrix}1 & 1\end{bmatrix}\; T^{F_n}(\,\cdot\,;\,\chi_{u_n},\,\chi^{v_n}) \cdots T^{F_2}(\,\cdot\,;\,\chi_{u_2},\,\chi^{v_2}) T^{F_1}(\,\cdot\,;\,\chi_{u_1},\,\chi^{v_1})\; \begin{bmatrix}1\\0\end{bmatrix}$$

If $c_i = 0$, then the matrices $T^{F_i}(\,\cdot\,;\,\chi_{u_i},\,\chi^{v_i})$ are given by

$$T^{F_i}(\,\cdot\,;\,\chi_0,\,\chi^0) = \frac{1}{2}\begin{bmatrix} 2 & 1\\ 0 & 1\end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\chi_1,\,\chi^0) = \frac{1}{2}\begin{bmatrix} 0 & 1\\ 0 & -1\end{bmatrix},$$
$$T^{F_i}(\,\cdot\,;\,\chi_0,\,\chi^1) = \frac{1}{2}\begin{bmatrix} 0 & -1\\ 0 & 1\end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\chi_1,\,\chi^1) = \frac{1}{2}\begin{bmatrix} 2 & -1\\ 0 & -1\end{bmatrix}.$$

If $c_i = 1$, then they are instead equal to

$$T^{F_i}(\,\cdot\,;\,\chi_0,\,\chi^0) = \frac{1}{2}\begin{bmatrix} 1 & 0\\ 1 & 2\end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\chi_1,\,\chi^0) = \frac{1}{2}\begin{bmatrix} 1 & 0\\ -1 & 0\end{bmatrix},$$
$$T^{F_i}(\,\cdot\,;\,\chi_0,\,\chi^1) = \frac{1}{2}\begin{bmatrix} -1 & 0\\ 1 & 0\end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\chi_1,\,\chi^1) = \frac{1}{2}\begin{bmatrix} -1 & 0\\ -1 & 2\end{bmatrix}.$$

The algorithm from Section A shows that these matrices are not simultaneously row- or column-monomializable. Hence, although the above formula provides an efficient algorithm to evaluate the correlations of linear approximations, we do not immediately obtain compact constraints.

Differential cryptanalysis. The formula for the correlations of quasidifferentials for the constant addition is obtained in an analogous way to Section 5.1. In total, we obtain 32 matrices of size 4×4. They are given in Appendix E. The algorithm from Section A demonstrates that these matrices cannot be simultaneously row- or column-monomialized.

Ultrametric integral cryptanalysis. Like in Section 5.1, the correlation of an ultrametric integral approximation with exponents u in $\mathbb{F}_2^n$ and v in $\mathbb{F}_2^n$ is given by

$$\begin{bmatrix}1 & 1\end{bmatrix}\; T^{F_n}(\,\cdot\,;\,\mu_{u_n},\,\mu^{v_n}) \cdots T^{F_2}(\,\cdot\,;\,\mu_{u_2},\,\mu^{v_2}) T^{F_1}(\,\cdot\,;\,\mu_{u_1},\,\mu^{v_1})\; \begin{bmatrix}1\\0\end{bmatrix}$$

If $c_i = 0$, then the matrices $T^{F_i}(\,\cdot\,;\,\mu_{u_i},\,\mu^{v_i})$ are given by

$$T^{F_i}(\,\cdot\,;\,\mu_0,\,\mu^0) = \begin{bmatrix} 1 & 1\\ 0 & 0\end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\mu_1,\,\mu^0) = \begin{bmatrix} 0 & -1\\ 0 & 1\end{bmatrix},$$
$$T^{F_i}(\,\cdot\,;\,\mu_0,\,\mu^1) = \begin{bmatrix} 0 & 1\\ 0 & 0\end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\mu_1,\,\mu^1) = \begin{bmatrix} 1 & -1\\ 0 & 0\end{bmatrix}.$$

If $c_i = 1$, then they are instead equal to

$$T^{F_i}(\,\cdot\,;\,\mu_0,\,\mu^0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\mu_1,\,\mu^0) = \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix},$$
$$T^{F_i}(\,\cdot\,;\,\mu_0,\,\mu^1) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \qquad T^{F_i}(\,\cdot\,;\,\mu_1,\,\mu^1) = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Modulo two, the above matrices are simultaneously column-monomializable. The change of basis transformation is given by the parity set basis (dual of the monomial basis). The resulting expression corresponds to the result obtained by Braeken and Semaev [15]. For ultrametric integral cryptanalysis, there does not exist a simultaneous row- or column-monomialization.

5.3 Chi function

This section shows how to compute the properties of the Chi function $\chi_n \colon \mathbb{F}_2^n \to \mathbb{F}_2^n$. A representation of χ_n as an S-function is shown in Figure 4. The state-update functions $F_i \colon \mathbb{F}_2^2 \times \mathbb{F}_2 \to \mathbb{F}_2^2 \times \mathbb{F}_2$ are all equal, and given by

$$F_i(\underbrace{x_{i-1} \| x_{i-2}}_{s_i}, x_i) = (\underbrace{x_i \| x_{i-1}}_{s_{i+1}},\ x_i + (x_{i-1}+1)x_{i-2})\,.$$

Note that the state s_i used to compute the i^{th} output bit consists of the previous two input bits x_{i-1} and x_{i-2}. The Chi function is an S-function with a cycle, i.e. $x_0 = x_n$ and $x_{-1} = x_{n-1}$. The results of Section 4.4 will be used to handle this.

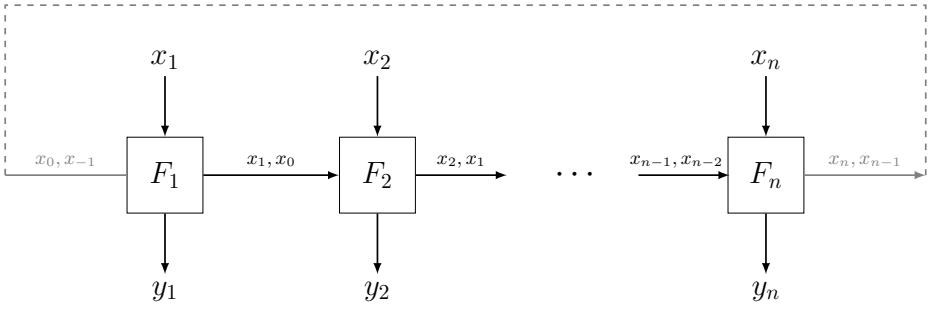


Fig. 4: The Chi function as an S-function. The dashed line indicates that the final state (x_n, x_{n-1}) is fed back as the initial state (x_0, x_{-1}), forming a cycle.

Linear cryptanalysis. As explained in Section 4.4, to handle the cyclic nature of the function, we need to force the initial and final states to equal values. If the input and output state are both fixed to s in $\mathbb{F}_2^2$, then $a = \delta_s$ and $b = \delta^s$ in

Theorem 2. Hence, the correlation of a linear approximation with masks u and v in $\mathbb{F}_2^n$ is equal to the sum

$$\sum_{s\in\mathbb{F}_2^2} \delta^s \Big(T^{F_n}(\,\cdot\,;\,\chi_{u_n},\,\chi^{v_n}) \cdots T^{F_2}(\,\cdot\,;\,\chi_{u_2},\,\chi^{v_2}) T^{F_1}(\,\cdot\,;\,\chi_{u_1},\,\chi^{v_1})\,\delta_s \Big)$$

Here, the four matrices $T^{F_i}(\,\cdot\,;\,\chi_{u_i},\,\chi^{u_i})$ with u_i and v_i in $\mathbb{F}_2$ are

$$T^{F_i}(\,\cdot\,;\,\chi_0,\,\chi^0) = \frac{1}{2}\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \quad T^{F_i}(\,\cdot\,;\,\chi_1,\,\chi^0) = \frac{1}{2}\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ -1 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 \end{bmatrix},$$

$$T^{F_i}(\,\cdot\,;\,\chi_0,\,\chi^1) = \frac{1}{2}\begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 \end{bmatrix}, \quad T^{F_i}(\,\cdot\,;\,\chi_1,\,\chi^1) = \frac{1}{2}\begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

The algorithm from Appendix A shows that these 4 matrices are not simultaneously row- or column-monomializable.

Differential cryptanalysis. Like in Sections 5.1 and 5.2, to compute the correlations of quasidifferentials, we should consider an extension of the state-update function that operates on pairs. Specifically, we consider

$$G_i(x_{i-1}\|x_{i-2}\|x'_{i-1}\|x'_{i-2}, x_i\|x'_i) = F_i(x_{i-1}\|x_{i-2}, x_i)\|F_i(x'_{i-1}\|x'_{i-2}, x'_i)\,.$$

Like in the linear case, the initial state and final state should be equal. Thus, for an input mask-difference pair (u, d) in $\mathbb{F}_2^n \times \mathbb{F}_2^n$ and an output mask-difference pair (v, e) in $\mathbb{F}_2^n \times \mathbb{F}_2^n$, the quasidifferential correlation is equal to

$$\sum_{s\in\mathbb{F}_2^4} \delta^s \Big(T^{F_n}(\,\cdot\,;\,q_{u_n,d_n},\,q^{v_n,e_n}) \cdots T^{F_2}(\,\cdot\,;\,q_{u_1,d_1},\,q^{v_1,e_1})\,\delta_s \Big)$$

In total we obtain 16 matrices of size 16×16 (see Appendix F). Algorithm 2 shows that these 16 matrices can not be simultaneously row- or column-monomialized.

Ultrametric integral cryptanalysis. Like in the linear cryptanalysis case, the correlation of an ultrametric integral approximation with exponents u in $\mathbb{F}_2^n$ and v in $\mathbb{F}_2^n$ is given by

$$\sum_{s\in\mathbb{F}_2^4} \delta^s \Big(T^{F_n}(\,\cdot\,;\,\mu_{u_n},\,\mu^{v_n}) \cdots T^{F_2}(\,\cdot\,;\,\mu_{u_2},\,\mu^{v_2}) T^{F_1}(\,\cdot\,;\,\mu_{u_1},\,\mu^{v_1})\delta_s \Big)$$

The matrices involved in the above formula are

$$T^{F_i}(\,\cdot\,;\,\mu_0,\,\mu^0) = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad T^{F_i}(\,\cdot\,;\,\mu_1,\,\mu^0) = \begin{bmatrix} -1 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix},$$

$$T^{F_i}(\,\cdot\,;\,\mu_0,\,\mu^1) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad T^{F_i}(\,\cdot\,;\,\mu_1,\,\mu^1) = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}.$$

Algorithm 2 shows that these four matrices are not simultaneously row- or column-monomializable, even if we take the elements modulo two.

5.4 Other examples of S-functions

The following examples are more complicated, but they demonstrate the generality of our results: as long as we can represent a function as an S-function, Theorem 2 and Algorithm 1 are applicable.

ChiChi function. The $\chi\!\!\chi$-function was introduced by Belkheyar et al. at Eurocrypt 2025 [6]. It is extended affine-equivalent to two parallel χ-functions, so its linear and differential properties can be computed from those in Section 5.3. However, this approach does not work for (ultrametric) integral cryptanalysis.

For consistency with [6] we use indices running from 0 to $n-1$ throughout this section. Let m be an even integer, and let $n = 2m$. The map $\chi\!\!\chi_n \colon \mathbb{F}_2 \to \mathbb{F}_2^n$ is defined by $(x_0, \ldots, x_{n-1}) \mapsto (y_0, \ldots, y_{n-1})$, where

$$y_i = \begin{cases} x_i + (x_{i+1}+1)x_{i+2} & i < m-3 \text{ or } m < i < n-2\,, \\ x_m + (x_{m-2}+1)x_0 & i = m-3\,, \\ x_{m-1} + (x_0+1)x_1 & i = m-2\,, \\ (x_{m-3}+1) + (x_m+1)(x_{m+1}+1) & i = m-1\,, \\ x_{m-2} + (x_{m+1}+1)x_{m+2} & i = m\,, \\ x_{n-2} + (x_{n-1}+1)x_{m-1} & i = n-2\,, \\ x_{n-1} + (x_{m-1}+1)x_m & i = n-1\,. \end{cases} \tag{1}$$

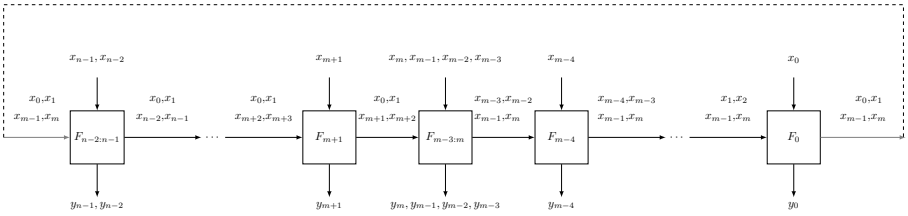


Fig. 5: S-function representation of the $\chi\!\!\chi_n$ function.

To use our framework to study the properties of $\chi\!\!\chi_n$, we need to represent it as an S-function. As shown in Figure 5, this is possible. The $n = 2m$ bit input and output bits are arranged into four types according to their position:

Type 1 For $i < m-3$, each state-update function is defined as $F_i\colon (x_{i+1}\|x_{i+2}\| x_{m-1}\|x_m, x_i) \mapsto (x_i\|x_{i+1}\|x_{m-1}\|x_m, y_i)$.

Type 2 For $m < i < n-2$, each state-update function has one input bit, one output bit, and four bits of input and output state. The state-update function is denoted by $F_i\colon (x_0\|x_1\|x_{i+2}\|x_{i+3}, x_i) \mapsto (x_0\|x_1\|x_{i+1}\|x_{i+2}, y_i)$.

Type 3 For $m-3 \le i \le m$, the middle four input bits are processed by a single state-update function, denoted by $F_{m-3:m}$ where

$$F_{m-3:m}\colon (x_0\|x_1\|x_{m+1}\|x_{m+2}, x_{m-3}\|x_{m-2}\|x_{m-1}\|x_m) \mapsto (x_{m-3}\|x_{m-2}\|x_{m-1}\|x_m, y_{m-3}\|y_{m-2}\|y_{m-1}\|y_m)\,.$$

Type 4 For $n-2 \le i \le n-1$, the last two bits are also regarded as a single function; the state-update function is $F_{n-2:n-1}$ as

$$F_{n-2:n-1}\colon (x_0\|x_1\|x_{m-1}\|x_m, x_{n-2}\|x_{n-1}) \mapsto (x_0\|x_1\|x_{n-2}\|x_{n-1}, y_{n-2}\|y_{n-1})\,.$$

The values y_i in the state-update functions above are defined in Equation (1).

Similar to the χ function, a cryptanalytic property for $\mathcal{X}_n$ evaluates to (e.g. in the case of linear or integral cryptanalysis)

$$\sum_{s\in\mathbb{F}_2^4} \delta^s \Big(M_0 \cdots M_{m-4} M_{m-3:m} M_{m+1} \cdots M_{n-3} M_{n-2:n-1}\ \delta_s \Big)\,.$$

In the above, for $0 \le i \le m-4$ or $m+1 \le i \le n-3$, $M_i = T^{F_i}(\,\cdot\,;\, u_i,\, v_i)$. For the other two state-update functions $M_{m-3:m} = T^{F_{m-3:m}}(\,\cdot\,;\, u_{m-3:m},\, v_{m-3:m})$ and $M_{n-2:n-1} = T^{F_{n-2:n-1}}(\,\cdot\,;\, u_{n-2:n-1},\, v_{n-2:n-1})$.

SHA-1 Step Function The step function of SHA-1 is as shown in Figure 6[10]. The nonlinear part of this step function can be represented by

$$\begin{aligned} A_{i+1} &= G'_i(A_i, B_i, C_i, D_i, E_i) = (A_i \lll 5) \boxplus F_i(B_i, C_i, D_i) \boxplus E_i \boxplus W_i \boxplus K_i \\ &= G_i(A'_i, B_i, C_i, D_i, E_i) \end{aligned} \tag{2}$$

$A_i, B_i, C_i, D_i, E_i \in \mathbb{F}_2^{32}$ are the chaining value of the i^{th} round, A_{i+1} is the chaining value of the $i+1$ round that is computed from the step function by a nonlinear function G'_i; Inside G'_i, $H_i : \mathbb{F}_2^{32\times 3} \to \mathbb{F}_2^{32}$ is a vectorial Boolean function according to the step index as

$$H_i(B_i, C_i, D_i) = \begin{cases} (B_i \wedge C_i) \vee ((\neg B_i) \wedge D_i) & 1 \le i \le 20, \\ B_i \oplus C_i \oplus D_i & 21 \le i \le 40, \\ (B_i \wedge C_i) \vee (B_i \wedge D_i) \vee (C_i \wedge D_i) & 41 \le i \le 60, \\ B_i \oplus C_i \oplus D_i & 61 \le i \le 80. \end{cases}$$

$W_i \in \mathbb{F}_2^{32}$ is the message absorbed into this step, and K_i is a constant.

[10] This figure is adapted from https://www.iacr.org/authors/tikz.

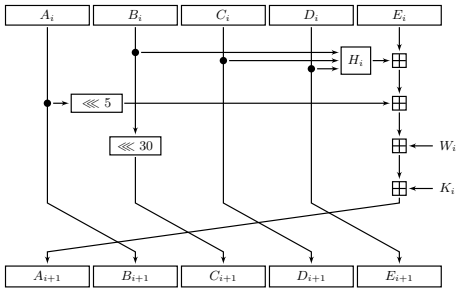


Fig. 6: The step function of SHA-1 hash function.

To compute the property for the step function G'_i, we should represent it as an S-function. For the sake of convenience, we will study G_i rather than G'_i in Equation 2 to avoid handling the rotations.

The first step is to split G_i as bit-wise state-update functions. Let $G_{i,w,k}$ be the state-update function in the i^{th} round with the constant w and k being bits of W_i and K_i, and let $f : \mathbb{F}_2^3 \to \mathbb{F}_2$ be the Boolean function used in this round. Since the state-update function contains 4 modular additions (with constants), we use 4 bits as the states. The state-update function is then

$$G_{i,w,k} : \mathbb{F}_2^4 \times \mathbb{F}_2^5 \to \mathbb{F}_2^4 \times \mathbb{F}_2; (\underbrace{c_0\|c_1\|c_2\|c_3}_{s_i}, a\|b\|c\|d\|e) \mapsto (\underbrace{t_0\|t_1\|t_2\|t_3}_{s_{i+1}}, y)$$

Here, t_i and y are given by

$$\begin{aligned}
y_0 &= f(b,c,d) + e + c_0 \\
t_0 &= f(b,c,d)e + f(b,c,d)c_0 + ec_0 \\
y_1 &= y_0 + a + c_1 \\
t_1 &= y_0 a + y_0 c_1 + ac_1 \\
y_2 &= y_1 + w + c_2 \\
t_2 &= y_1 w + y_1 c_2 + w_i c_2 \\
y &= y_2 + k + c_3 \\
t_3 &= y_2 k + y_2 c_3 + k c_3
\end{aligned}$$

According to the concrete values w, k, and the Boolean function f, the algebraic normal form of $G_{i,w,k}$ can be determined, with y_0, y_1, y_2 as auxiliary variables. The following steps are similar to those in other examples in this section.

6 Boomerang Distinguisher for Subterranean 2.0

In this section, we present a boomerang distinguisher for Subterranean 2.0 [20]. However, the data required to exploit this distinguisher is higher than the security claim for any construction based on the permutation. Hence, the main goal

of this section is to demonstrate how our results can be used to analyze more complex, combined properties of primitives based on S-functions.

Subterranean 2.0 uses χ_{257} as its sole non-linear function and has been used as the underlying permutation of Subterranean-SAE, one of the second round candidates of the NIST LWC competition[11]. As is well-known, the inverse of a large χ-function is complicated, resulting in significant difficulty in computing the Boomerang Connectivity Table (BCT) for χ_{257}. Recall that BCT is defined as follows.

Definition 4 (Boomerang connectivity table [17]). *Let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an invertible function. The Boomerang Connectivity Table of F is given by a $2^n \times 2^n$ table BCT_F, in which the entry for the $(\Delta, \nabla) \in \mathbb{F}_2^n \times \mathbb{F}_2^n$ position is given by*

$$BCT_F[\Delta, \nabla] = 2^{-n} \#\{x \in \mathbb{F}_2^n \mid F^{-1}(F(x) + \nabla) + F^{-1}(F(x + \Delta) + \nabla) = \Delta\}\,.$$

This is the probability that the boomerang with differences (Δ, ∇) returns.

Subterranean 2.0 consists of 8 rounds. The i-th round function ρ_i, is defined as $\rho_i = \pi \circ \theta \circ \iota_i \circ \chi_{257}$, where ι_i is a constant addition operation, and θ and π are two linear operations.

6.1 Computing the BCT of χ_{257} using 3-differentials

Kidmose and Tiessen [25, Theorem 2] linked the probability of boomerangs to truncated 3-differentials [42]. Formally, the relation is as follows.

Theorem 3. *Let $A = \{(\Delta, a, \Delta + a) \mid a \in \mathbb{F}_2^n\}$ be the affine subspace of all 3-differences corresponding to an input quartet. Likewise, let $B = \{(b, \nabla, b + \nabla) \mid b \in \mathbb{F}_2^n\}$ be the set of all 3-differences corresponding to an output quartet. The probability that the boomerang with differences (Δ, ∇) returns, is equal to*

$$BCT_F[\Delta, \nabla] = 2^n \Pr\big(A \xrightarrow{F} B\big)\,,$$

where the right-hand side is the probability of the truncated 3-differential (A, B).

Hence, to compute the BCT, we compute the truncated 3-differential probability

$$\Pr\big(A \xrightarrow{F} B\big) = \frac{1}{2^n} \sum_{a \in \mathbb{F}_2^n} \sum_{b \in \mathbb{F}_2^n} \Pr\big((\Delta, a, \Delta + a) \xrightarrow{F} (b, \nabla, b + \nabla)\big)\,.$$

This probability could be computed directly using Theorem 2, but we opt for a more indirect approach to demonstrate that we can also calculate the probabilities of 3-differentials over S-functions. In [44], the notion of quasi-3-differentials was proposed as a direct extension of the quasidifferential framework [10]. Let X be the set of quartets, i.e. $X = \mathbb{F}_2^n \times \mathbb{F}_2^n \times \mathbb{F}_2^n \times \mathbb{F}_2^n$. This also means that the primitive F should be extended to a function on quartets $(x, y, z, w) \mapsto$

[11] https://csrc.nist.gov/Projects/Lightweight-Cryptography

Table 2: Four-round differential characteristic with a probability 2^{-58} from [20].

Difference	Weight	Active bit indices
a_1	12	$\{0, 5, 8, 10, 12, 15, 16, 18, 21\}$
b_1	7	$\{65, 66, 85, 86, 87\}$
b_2	11	$\{7, 28, 134, 198, 200, 219\}$
b_3	28	$\{16, 18, 22, 39, 54, 86, 88, 107, 118, 139, 152, 173, 188, 211, 252\}$
b_4	—	$\{3, 12, 15, 21, 23, 24, 31, 33, 39, 44, 52, 57, 78, 85, 93, 97, 103, 106$ $116, 118, 124, 130, 133, 157, 178, 180, 184, 187, 194, 196, 197, 201,$ $215, 216, 218, 224, 248, 250, 251\}$

$(F(x), F(y), F(z), F(w))$. The basis for $\mathbb{R}^X$ introduced in [44] is called the quasi-3-differential basis. It consists of the functions

$$q^{(u,d,e,f)}(x, y, z, w) = (-1)^{u^\top x}\, \delta_d(x + y)\delta_e(x + z)\delta_f(x + w)\,.$$

The dual basis for $\mathbb{R}[\mathbb{F}_2^n]$ consists of the vectors $p_{(u,d,e,f)}$ with coordinates

$$q_{(u,d,e,f)}[(x, y, z, w)] = (-1)^{u^\top x}\, \delta_d(x + y)\delta_e(x + z)\delta_f(x + w)/2^n\,.$$

Using the same approach that we used to compute the quasidifferential properties for χ_n in Section 5.3, we can compute the correlations of quasi-3-differentials. The state consists of a 2-bit value and three 2-bit differences, i.e. a total of eight bits. Hence, there exist 256×256 matrices $M_1, \ldots, M_n$ such that

$$\Pr\big((\Delta, a, \Delta + a) \xrightarrow{\chi_n} (b, \nabla, b + \nabla)\big) = \sum_{s\in\mathbb{F}_2^8} \delta^s\big(M_{a_n,b_n} \cdots M_{a_2,b_2} M_{a_1,b_1}\, \delta_s\big)\,.$$

In particular, $M_{a_i,b_i} = T^{F_i}(\,\cdot\,;\, q_{(0,\Delta_i,a_i,\Delta_i+a_i)},\, q^{(0,b_i,\nabla_i,b_i+\nabla_i)})$, where F_i is the state-update function for the quartet-extended χ-function. Consequently,

$$\begin{aligned} 2^n \Pr\big(A \xrightarrow{\chi_n} B\big) &= \sum_{a\in\mathbb{F}_2^n} \sum_{b\in\mathbb{F}_2^n} \sum_{s\in\mathbb{F}_2^8} \delta^s\big(M_{a_n,b_n} \cdots M_{a_1,b_1}\delta_s\big) \\ &= \sum_{s\in\mathbb{F}_2^8} \delta_s \left(\left(\textstyle\sum_{a_n,b_n\,\in\,\mathbb{F}_2} M_{a_n,b_n}\right) \cdots \left(\textstyle\sum_{a_1,b_1\in\mathbb{F}_2} M_{a_1,b_1}\right) \delta_s\right)\,. \end{aligned}$$

Finally, $BCT_F[\Delta, \nabla] = 2^n \Pr\big(A \xrightarrow{\chi_n} B\big)$ is obtained.

6.2 Boomerang distinguisher for 8-round Subterranean 2.0

In [20], the authors report a 4-round differential characteristic with probability 2^{-58}, as shown in Table 2. Let b_i and a_i be the differences before and after the i^{th}-round χ_{257} operation, i.e., the characteristic is

$$b_0 \xrightarrow{\chi_{257}} a_1 \xrightarrow{\pi\circ\theta\circ\iota_1} b_1 \xrightarrow{\chi_{257}} a_2 \xrightarrow{\pi\circ\theta\circ\iota_2} b_2 \xrightarrow{\chi_{257}} a_3 \xrightarrow{\pi\circ\theta\circ\iota_3} b_3 \xrightarrow{\chi_{257}} a_4 \xrightarrow{\pi\circ\theta\circ\iota_4} b_4\,.$$

The difference b_0 is not explicitly specified in Table 2, but it can be any value as long as the probability over the first χ_{257} function is 2^{-12}. This is possible for the given difference a_1. More details can be found in [20].

We take this four-round characteristic as the upper differential characteristic. The lower differential characteristic is chosen as the three-round differential characteristic from b_4 to a_1, whose probability is 2^{-46}. The BCT at differences b_4 and a_1 is used to connect the upper and the lower characteristic, to obtain an 8-round boomerang distinguisher.

Given the upper difference b_4 and the lower difference a_1, the probability that the boomerang returns is $2^{-5.8}$. Hence, the overall boomerang probability is estimated as $2^{-(5.8+58\times 2+46\times 2)} = 2^{-213.8}$. This is a boomerang distinguisher for the full Subterranean 2.0 permutation. Although this distinguisher clearly does not affect the security of Subterranean-SAE or other constructions based on Subterranean 2.0, being able to compute the probability of boomerangs is of interest for designers who want to use large S-functions such as χ_{257}.

7 Conditional Linear Approximations and Partitioning

The partitioning technique was introduced by Biham and Carmeli [14] to improve a linear attack against FEAL-8X. The basic idea is to partition the domain of a primitive into several smaller sets and to evaluate a linear approximation on the restriction of the primitive to each of these sets separately.

Later, partitioning was applied by Leurent [29] to improve differential-linear attacks on Chaskey [36]. Beierle et al. [4, 5] extended the technique to two consecutive modular additions to improve these attacks. At Asiacrypt 2023, Chen et al. [16] further advanced the method by applying it to three parallel modular additions, thereby improving key-recovery attacks against LEA [21].

However, when the authors of [4, 5] and [16] used the partitioning technique, they unfortunately could not compute exact (conditional) correlations for the complicated functions involving consecutive modular additions. Hence, they relied on a combination of experimental estimates and the piling-up lemma to approximate the correlation. This approach came with the risk of introducing inaccuracies. In this section, we show that our results can be used to calculate the correlations exactly.

7.1 Revisiting the partitioning technique in [16]

We begin by abstracting the core problem in [16], without giving the full context.

Figure 7 shows a part of the round function of LEA, where z_i, k_i and y_i are in $\mathbb{F}_2^{32}$. Let $\Omega = (z_0, k_0, k_1, k_2, y_0, y_1, y_2) \in \mathbb{F}_2^{7\cdot 32}$. The authors of [16] first partition $\mathbb{F}_2^{7\cdot 32}$, into 2^{12} small sets based on 12 equations that are listed in Appendix B. The sets are parameterized by 12 bits $v_1, v_2, \ldots, v_{12}$.

Let $P_1, \ldots, P_{2^{12}}$ the sets in the partition, so that $\mathbb{F}_2^{7\cdot 32} = \bigcup_{i=1}^{2^{12}} P_i$. Let γ be a linear mask in $\mathbb{F}_2^{7\cdot 32}$. One then computes the correlation of the linear approximation $z_1[27] + z_2[29] + z_3[9] = \gamma^\top \Omega$ on each space P_i. However, at first sight,

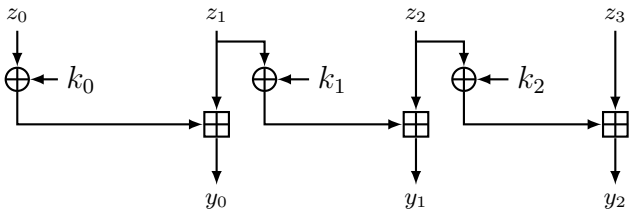


Fig. 7: Part of the LEA round function to which partitioning is applied [16].

there is no simple way to compute the correlation of this approximation over the structure in Figure 7. For this reason, the authors of [16] conduct experiments to compute the correlations for each of the following three linear approximations

$$z_1[27] = {\gamma_1}^\top \Omega, \quad z_2[29] = {\gamma_2}^\top \Omega, \quad z_3[9] = {\gamma_3}^\top \Omega\,,$$

where $\gamma_1 + \gamma_2 + \gamma_3 = \gamma$. The masks $\gamma_0, \gamma_1, \gamma_2$ are specifically chosen to make the experiments feasible. The correlation of $z_1[27] + z_2[29] + z_3[9] = \gamma^\top \Omega$ is then estimated by multiplying the correlations of these three linear approximations. The authors motivate this by the piling-up lemma, i.e. the assumption that these three approximations are independent.

7.2 Method to compute the exact conditional correlations

Given an input mask for $(z_0, z_1, z_2, z_3, k_0, k_1, k_2)$ and an output mask for (y_0, y_1, y_2), we want to compute the correlation of the corresponding linear approximation over Figure 7, restricted to each set P_i.

Since the construction in Figure 7 consists of only xors and modular additions, it is an S-function. Inspecting the 12 conditions in Appendix B, all constraints are either on one coordinate of the variables (e.g., $v_1 = (y_0 + z_0 + k_0)[8]$) or on two coordinates (e.g., $v_4 = (y_1 + k_1)[8] + y_0[7]$ if $v_2 = 1$). In the latter case, we can further partition the set P_i into smaller sets by fixing one of the two bits. For example, we can set $y_0[7] = 0$ for $v_4 = (y_1 + k_1)[8] + y_0[7]$ and then the value of $(y_1 + k_1)[8]$ is also fixed. In this way, all conditions are bitwise. This is important when we define the bitwise state-update function of our S-function.

If a subspace P_i is further partitioned into smaller subspaces, i.e. $P_i = \bigcup_j P_{i,j}$, then we can apply Algorithm 1 to compute the correlation for the restriction to each set $P_{i,j}$. The correlation for the restriction to P_i is then the weighted average of the correlations computed for all $P_{i,j}$.

There are three modular additions, so the state of the S-function can be described with three bits. To incorporate the conditions, we define the state-update function $F_i \colon \mathbb{F}_2^3 \times \mathbb{F}_2^7 \to \mathbb{F}_2^3 \times \mathbb{F}_2^3$ as follows:

$$F_i(s_i, z_0, z_1, z_2, z_3, k_0, k_1, k_2) = \begin{cases} (s_{i+1}, y_0, y_1, y_2) & \text{if the conditions is satisfied}\,, \\ \bot & \text{otherwise}\,. \end{cases}$$

The truth table of F_i can easily be computed. Hence, we can call Algorithm 1 to compute the desired correlation for any given input and output mask.

In this example, the correlation is actually computed in a subset P_i (or $P_{i,j}$) of $\mathbb{F}_2^{7\cdot 32}$. Hence, directly using the character basis from Section 2.2 would yield a correlation equal to the true correlation multiplied by $|P_i|/2^{7\cdot 32}$. To recover the actual correlation, we must rescale the result by a factor $2^{7\cdot 32}/|P_i|$. The size of P_i can be obtained from the correlation for the approximation with input and output mask zero.

7.3 More precise correlations for the LEA round function

Using the approach from Section 7.2, we can revisit the results of [16]. As a first example, consider the linear approximation that is the sum of the three approximations in Table 3, under the intersection of the three conditions. Under the assumption that the three linear approximations are independent, the authors multiplied the three correlations — resulting in a correlation of $-3/4$. However, our approach shows that this linear approximation actually has correlation -1.

To check the approximations used in Section 7.2, we have computed the correlation of all linear approximations that are used in [16]. When $(v_{12}, v_9) \notin \{(0,1),(1,1)\}$, the estimates from [16] are consistent with our theoretical values, thus correct. However, for $(v_{12}, v_9) \in \{(1,0),(0,0)\}$, their estimates differ from the correct values. In Appendix C, we list two figures that show the gap between the estimates from [16] and the correct values. The gap implies that the complexities and success probabilities of the attacks in [16] should be reconsidered. Although the precise impact is difficult to assess, we expect that modifying the LLR statistic used for the key-recovery attack in [16] according to the updated values of the correlations would result in a higher filter probability. Additional details are given in Appendix C.

For the attack on Chaskey in [5], we also reevaluated the correlations of the conditional linear approximation estimated using the piling-up lemma. Although discrepancies exist in many places, the maximum difference does not exceed 0.0002. Given that the smallest correlation among these conditional linear approximations is 0.40625, this has no impact on the key recovery process.

8 Inverse of the Chi Function

Daemen [19, §6.6] showed that the χ_n-function is invertible for odd n by constructing an algorithm to compute its inverse. However, this algorithm does not reveal the algebraic normal form of χ_n^{-1}. A theoretical method to compute the algebraic normal form of χ_n^{-1} was only recently proposed by Liu et al. [32]. In this section, we propose a general method to compute the algebraic normal form of the inverse of an S-function. This provides an alternative approach to obtain the results of Liu et al. [32].

Table 3: Three linear approximations under conditions in [16].

Approximation	Condition	Estimated correlation [16]
$z_1[27] = y_0[27] + z_0[27]$ $+z_0[25] + k_0[27] + k_0[25]$	$v_{12} = 1$	1
$z_2[29] = y_1[29] + y_1[26]$ $+y_0[29] + z_0[29] + z_0[27]$ $+k_1[29] + k_0[29] + k_0[27]$	$(v_7, v_8, v_9, v_{10}, v_{11})$ $= (0, 1, 0, 0, 1)$	$-3/4$
$z_3[9] = y_2[9] + y_2[8]$ $+y_1[9] + y_1[8] + y_0[9]$ $+z_0[9] + z_0[6] + k_2[9]$ $+k_1[9] + k_0[9] + k_0[6]$	$(v_1, v_2, v_3, v_4, v_5, v_6)$ $= (0, 0, 1, 0, 0, 1)$	1

8.1 Integral properties of χ_n^{-1}

The integral properties of χ_n were already analyzed in Section 5.3. Since χ_n is a bijection, T^{χ_n} is a permutation matrix and $T^{\chi_n^{-1}} = \left(T^{\chi_n}\right)^\top$ [7, Theorem 3.1]. By Lemma 1, the S-function representation $F = \text{S-FUNCTION}[F_1, \ldots, F_n]$ of χ_n satisfies $T^{F(\,\cdot\,; x, y)} = T^{F_n(\,\cdot\,; x_n, y_n)} \cdots T^{F_1(\,\cdot\,; x_1, y_1)}$. As transposition reverses the order of a product, it follows that

$$T^{F^{-1}(\,\cdot\,; x, y)} = \left(T^{F(\,\cdot\,; x, y)}\right)^\top = \left(T^{F_1(\,\cdot\,; x_1, y_1)}\right)^\top \cdots \left(T^{F_n(\,\cdot\,; x_n, y_n)}\right)^\top.$$

Here $T^{F_i^{-1}}$ abbreviates $\left(T^{F_i}\right)^\top$. The blocks F_i are not invertible, so this is the transition matrix of their inverse relation; it represents χ_n^{-1} only globally, because χ_n is a permutation. In combination with Theorem 2, for all $a \otimes u \in \mathbb{K}[\mathbb{F}_2^2 \times \mathbb{F}_2^n]$ and $b \otimes v \in \mathbb{K}^{\mathbb{F}_2^2 \times \mathbb{F}_2^n}$,

$$(b \otimes v)\Big(T^{F^{-1}}(a \otimes u)\Big) = b\Big(T^{F_1^{-1}}(\,\cdot\,;\, u_1,\, v_1) \cdots T^{F_n^{-1}}(\,\cdot\,;\, u_n,\, v_n)\, a\Big). \tag{3}$$

By choosing u and v to be vectors from the ultrametric integral basis and its dual as in Section 5.3, we derive the matrices necessary to compute the ultrametric integral properties of F^{-1}. By taking each element of these matrices modulo two, we obtain matrices that can be used to compute integer properties or, equivalently, monomial predictions [11,22]. We obtain the following four matrices over $\mathbb{F}_2$:

$$T^{F_i^{-1}}(\,\cdot\,;\, \mu_0,\, \mu^0) = \begin{bmatrix} 1&0&0&0\\ 0&0&1&0\\ 0&1&0&0\\ 0&1&0&0 \end{bmatrix}, \quad T^{F_i^{-1}}(\,\cdot\,;\, \mu_1,\, \mu^0) = \begin{bmatrix} 1&0&1&0\\ 1&0&1&0\\ 0&1&0&1\\ 0&1&0&1 \end{bmatrix},$$

$$T^{F_i^{-1}}(\,\cdot\,;\,\mu_0,\,\mu^1) = \begin{bmatrix} 0&0&0&0\\ 0&0&1&0\\ 0&0&0&0\\ 0&0&0&0 \end{bmatrix}, \quad T^{F_i^{-1}}(\,\cdot\,;\,\mu_1,\,\mu^1) = \begin{bmatrix} 0&0&1&0\\ 0&0&1&0\\ 0&0&0&1\\ 0&0&0&1 \end{bmatrix}.$$

For brevity, for exponents u and v in $\mathbb{F}_2^n$, let $M_{v_i,u_i} = T^{F_i^{-1}}(\,\cdot\,;\,\mu_{u_i},\,\mu^{v_i})$. By Equation (3), and taking into account the fact that χ_n^{-1} is a cyclic S-function, the coefficient of the monomial x^u in the algebraic normal form of $(\chi_n^{-1})^v$ is

$$c_{v,u} = \sum_{s\in\mathbb{F}_2^2} \delta^s\big(M_{v_1,u_1}\cdots M_{v_n,u_n}\delta_s\big) = \mathrm{Tr}\big(M_{v_1,u_1}\cdots M_{v_n,u_n}\big)\,, \tag{4}$$

where all sums are in $\mathbb{F}_2$.

8.2 Computing the algebraic normal form of χ_n^{-1}

To compute the algebraic normal form of $\chi_n^{-1} : y \mapsto x$, we need to find out all monomials of y^u that appears in each coordinate polynomial of x. Let v be a unit vector whose i-th ($1 \le i \le n$) coordinate is 1. For a monomial of y^u, according to Equation (4), $c_{v,u} = Tr(M_{v_1,u_1}\cdots M_{v_n,u_n})$ determines whether y^u appears in x^v (denoted by $y^u \to x^v$ in the following). Due to the cyclic property of the trace, we have

$$Tr(M_{v_1,u_1}\cdots M_{v_n,u_n}) = Tr(M_{v_{i+1},u_{i+1}}\cdots M_{v_n,u_n}M_{v_1,u_1}\cdots M_{v_i,u_i})$$

thus if $y^u \to x^v$, $y^{u\lll i} \to x^{v\lll i}$. This means that the algebraic normal forms of χ_n^{-1} also have a shift-invariant property.

Therefore, we only need to compute the algebraic normal form of one coordinate. Hence, without loss of generality, we can compute the least significant bit, i.e., the algebraic normal form of $(\chi_n^{-1})^v$ with $v = (0,0,\ldots,0,1)$, i.e., $x^{(0,0,\ldots,0,1)} = x_n$.

Determining the algebraic normal form is equivalent to computing all exponents u such that $c_{v,u} = 1$ in Equation (4), i.e., so that x^u occurs in the algebraic normal form. Let us call these exponents u *valid*. Since $v = (0,\ldots,1)$, finding all valid u is equivalent to determining all possible sequences of matrices

$$M_{0,u_1},\ldots,M_{0,u_{n-1}},M_{1,u_n}\,,$$

so that Equation (4) holds. It is convenient to translate this problem into a path-finding problem in a directed graph that represents a finite state machine. Let us define the 'start' state as the matrix $[\delta_{00},\delta_{01},\delta_{10},\delta_{11}]$. Multiplying this matrix with one of the matrices $M_{0,0}$, $M_{0,1}$ or $M_{1,0}$, $M_{1,1}$ ($M_{1,0}$ and $M_{1,1}$ are applied only in the first step, since only $v_n = 1$) yields another matrix corresponding to a new state.

Let $1 \le j \le n$ and define

$$S^j = (S_0^j, S_1^j, S_2^j, S_3^j) = M_{0,u_{n+1-j}}\cdots M_{0,u_{n-1}}\cdot M_{1,u_n}\cdot \mathrm{start}$$

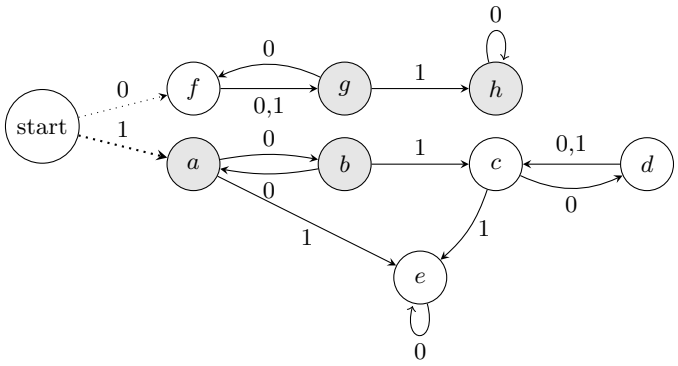


Fig. 8: The finite state machine for the transition of states of χ_n^{-1}.

$M_{0,u_{n+1-j}} \cdots M_{0,u_{n-1}} \cdot M_{1,u_n}$ can be seen as a path that leads start node to S^j. By enumerating possible $u_n, u_{n-1}, \ldots$, the possible values of S^j converge to a set containing only 8 non-zero states (so we do not need to really enumerate all 2^n values of $u_n, u_{n-1}, \ldots u_1$). Finiteness is automatic: every S^j is a product of 4×4 matrices over $\mathbb{F}_2$, of which there are only finitely many. Closing the orbit of the two first-step states $f = M_{1,0} \cdot \text{start}$ and $a = M_{1,1} \cdot \text{start}$ under left multiplication by the two generators $M_{0,0}$ and $M_{0,1}$ terminates at the eight nonzero states listed below (the absorbing all-zero matrix is discarded). The 8 states are as follows,

$$a = \begin{bmatrix} 0&0&1&0 \\ 0&0&1&0 \\ 0&0&0&1 \\ 0&0&0&1 \end{bmatrix} \quad b = \begin{bmatrix} 0&0&1&0 \\ 0&0&0&1 \\ 0&0&1&0 \\ 0&0&1&0 \end{bmatrix} \quad c = \begin{bmatrix} 0&0&0&0 \\ 0&0&0&0 \\ 0&0&1&1 \\ 0&0&1&1 \end{bmatrix} \quad d = \begin{bmatrix} 0&0&0&0 \\ 0&0&1&1 \\ 0&0&0&0 \\ 0&0&0&0 \end{bmatrix}$$

$$e = \begin{bmatrix} 0&0&1&1 \\ 0&0&1&1 \\ 0&0&1&1 \\ 0&0&1&1 \end{bmatrix} \quad f = \begin{bmatrix} 0&0&0&0 \\ 0&0&1&0 \\ 0&0&0&0 \\ 0&0&0&0 \end{bmatrix} \quad g = \begin{bmatrix} 0&0&0&0 \\ 0&0&0&0 \\ 0&0&1&0 \\ 0&0&1&0 \end{bmatrix} \quad h = \begin{bmatrix} 0&0&1&0 \\ 0&0&1&0 \\ 0&0&1&0 \\ 0&0&1&0 \end{bmatrix}$$

Using the 8 states of S^j and the start node as vertices, and M_{v_i,u_i} as edges, we can form the state transition graph shown in Figure 8. The edge 0 means $M_{1,0}$ or $M_{0,0}$; The edge 1 means $M_{1,1}$ or $M_{0,1}$. Equation (4) yields $c_{v,u} = 1$ only when the path corresponding to u ends in a state with trace one, i.e.,

$$c_{v,u} = (\delta^{00}, \delta^{01}, \delta^{10}, \delta^{11}) S^n = 1.$$

Then only $S^n \in \{a, b, g, h\}$ satisfies this requirement. In other words, a valid path must end at a, b, g or h. To determine the valid paths, we split the analysis into two cases.

$u_n = 1$: lower part of Figure 8. If a path reaches state c, d or e, then it can never return to a or b. However, the path must end in either a or b. It

follows that there is a unique valid path, corresponding to $u = (0, 0, 0, \ldots, 1)$. Hence, this case only contributes the monomial y_n to the algebraic normal form of $y \mapsto \chi_n^{-1}(y)^v$.

$u_n = 0$: upper part of Figure 8. If a path reaches h, it will never return and will end at h in the cycle with edge zero. An accepting path in the upper part must reach h: while the state alternates between f and g it is f at the odd positions $S^1, S^3, \ldots$ and g at the even positions, so for odd n we have $S^n \neq g$; only the h-loop yields an accepting state at position n. Before that happens, the state can alternate between f and g an arbitrary number of times. Hence, all valid paths take the following form (for $1 \leq j \leq \frac{n-1}{2}$):

$$\text{start} \xrightarrow{0} \underbrace{f \underset{0}{\overset{0,1}{\rightleftarrows}} g}_{j \geq 1 \text{ times}} \xrightarrow{1} h \underbrace{\xrightarrow{0} h \cdots h \xrightarrow{0}}_{n-1-2j \text{ times}} h\,,$$

where j is the number of times from f to g. These paths correspond to the following sum of monomials y^u, where u is a valid exponent:

$$\sum_{u_{n-1},\ldots,u_{n-2j+1}} y_{n-1}^{u_{n-1}}\, y_{n-3}^{u_{n-3}} \cdots y_{n-2j+1}^{u_{n-2j+1}}\, y_{n-2j} = y_{n-2j} \prod_{k=1}^{j} (y_{n-2k+1} + 1)\,.$$

After summing the valid monomials from the upper and lower parts of Figure 8, and using the shift-invariance of χ_n^{-1}, we obtain the result,

$$x_i = y_i + \sum_{j=1}^{(n-1)/2} y_{i-2j} \prod_{k=1}^{j} (y_{i-2k+1} + 1).$$

The same approach can be used for other S-functions, although there is no guarantee that the algebraic normal form is as simple as for χ_n^{-1}.

Acknowledgments. The authors thank the anonymous reviewers for their valuable comments. This work is supported by the National Key R&D Program of China (Grant No. 2024YFA1013000). Zhongfeng Niu is supported by the Singapore NRF-NRFI08-2022-0013 grant. Tim Beyne is supported by a postdoctoral fellowship from the Research Foundation – Flanders (FWO) with reference number 1274724N. Kai Hu and Meiqin Wang are supported by the National Natural Science Foundation of China (Grant No. U2336207, 62032014), the Department of Science & Technology of Shandong Province (No. SYS202201), and the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM114). Kai Hu is also supported by the National Cryptologic Science Fund of China (2025NCSF02007), the National Natural Science Foundation of China (62402283), the Shandong Provincial Natural Science Foundation (No. 2025HWYQ-025), the Natural Science Foundation of Jiangsu Province (BK20240420), the Program of TaiShan Scholars Special Fund for young scholars (Grant No. tsqn202507063), and the Program of Qilu Young Scholars of Shandong University. Meiqin Wang is further supported by the National Cryptologic Science Fund of China (2025NCSF01013).

References

1. Azimi, S.A., Ranea, A., Salmasizadeh, M., Mohajeri, J., Aref, M.R., Rijmen, V.: A bit-vector differential model for the modular addition by a constant. In: ASIACRYPT 2020. LNCS, vol. 12491, pp. 385–414. Springer (2020) 3, 17
2. Azimi, S.A., Ranea, A., Salmasizadeh, M., Mohajeri, J., Aref, M.R., Rijmen, V.: A bit-vector differential model for the modular addition by a constant and its applications to differential and impossible-differential cryptanalysis. Des. Codes Cryptogr. **90**(8), 1797–1855 (2022) 3
3. Bar-On, A., Dunkelman, O., Keller, N., Weizman, A.: DLCT: A new tool for differential-linear cryptanalysis. In: EUROCRYPT 2019. LNCS, vol. 11476, pp. 313–342. Springer (2019) 2
4. Beierle, C., Broll, M., Canale, F., David, N., Flórez-Gutiérrez, A., Leander, G., Naya-Plasencia, M., Todo, Y.: Improved differential-linear attacks with applications to ARX ciphers. J. Cryptol. **35**(4), 29 (2022) 26, 40
5. Beierle, C., Leander, G., Todo, Y.: Improved differential-linear attacks with applications to ARX ciphers. In: CRYPTO 2020. LNCS, vol. 12172, pp. 329–358. Springer (2020) 26, 28
6. Belkheyar, Y., Derbez, P., Ghosh, S., Leander, G., Mella, S., Perrin, L., Rasoolzadeh, S., Stennes, L., Sun, S., Assche, G.V., Vizár, D.: Chilow and chichi: New constructions for code encryption. In: EUROCRYPT 2025. LNCS, vol. 15601, pp. 212–243. Springer (2025) 4, 21
7. Beyne, T.: A geometric approach to linear cryptanalysis. In: ASIACRYPT 2021. LNCS, vol. 13090, pp. 36–66. Springer (2021) 3, 4, 5, 29
8. Beyne, T.: A Geometric Approach to Symmetric-Key Cryptanalysis. Ph.D. thesis, KU Leuven (2023) 3, 4, 5, 6, 7
9. Beyne, T.: Some comments on commutative diagram cryptanalysis (2024), presentation at the Lorentz workshop 'Beating Real-Time Crypto: Solutions and Analysis' 5
10. Beyne, T., Rijmen, V.: Differential cryptanalysis in the fixed-key model. In: CRYPTO 2022. LNCS, vol. 13509, pp. 687–716. Springer (2022) 2, 5, 6, 15, 16, 24
11. Beyne, T., Verbauwhede, M.: Integral cryptanalysis using algebraic transition matrices. IACR Trans. Symmetric Cryptol. **2023**(4), 244–269 (2023) 5, 29
12. Beyne, T., Verbauwhede, M.: Ultrametric integral cryptanalysis. In: ASIACRYPT 2024. LNCS, vol. 15490, pp. 392–423. Springer (2024) 5
13. Beyne, T., Verbauwhede, M.: Integral cryptanalysis in characteristic p. In: ASIACRYPT 2025. LNCS, vol. 16245, pp. 66–96. Springer (2025) 2, 6
14. Biham, E., Carmeli, Y.: An improvement of linear cryptanalysis with addition operations with applications to FEAL-8X. In: SAC 2014. LNCS, vol. 8781, pp. 59–76. Springer (2014) 26
15. Braeken, A., Semaev, I.A.: The ANF of the composition of addition and multiplication mod 2^n with a boolean function. In: FSE 2005. LNCS, vol. 3557, pp. 112–125. Springer (2005) 3, 15, 17, 19
16. Chen, Y., Bao, Z., Yu, H.: Differential-linear approximation semi-unconstrained searching and partition tree: Application to LEA and speck. In: ASIACRYPT 2023. LNCS, vol. 14440, pp. 223–255. Springer (2023) 4, 26, 27, 28, 29, 38, 39, 40
17. Cid, C., Huang, T., Peyrin, T., Sasaki, Y., Song, L.: Boomerang connectivity table: A new cryptanalysis tool. In: EUROCRYPT 2018. LNCS, vol. 10821, pp. 683–714. Springer (2018) 2, 24

18. Daemen, J.: Cipher and hash function design strategies based on linear and differential cryptanalysis. Ph.D. thesis, Doctoral Dissertation, March 1995, KU Leuven (1995) 2
19. Daemen, J.: Cipher and hash function design strategies based on linear and differential cryptanalysis. Ph.D. thesis, KU Leuven (1995) 28
20. Daemen, J., Massolino, P.M.C., Mehrdad, A., Rotella, Y.: The subterranean 2.0 cipher suite. IACR Trans. Symmetric Cryptol. **2020**(S1), 262–294 (2020) 23, 25, 26
21. Hong, D., Lee, J., Kim, D., Kwon, D., Ryu, K.H., Lee, D.: LEA: A 128-bit block cipher for fast encryption on common processors. In: WISA 2013. LNCS, vol. 8267, pp. 3–27. Springer (2013) 2, 26
22. Hu, K., Sun, S., Wang, M., Wang, Q.: An algebraic formulation of the division property: Revisiting degree evaluations, cube attacks, and key-independent sums. In: ASIACRYPT 2020. LNCS, vol. 12491, pp. 446–476. Springer (2020) 29
23. Hu, K., Yap, T.: Perfect monomial prediction for modular addition. IACR Trans. Symmetric Cryptol. **2024**(3), 177–199 (2024) 3, 15, 17
24. Hu, K., Zhang, C., Chang, C., Zhang, J., Wang, M., Peyrin, T.: Unlocking mix-basis potential: Geometric approach for combined attacks. In: CRYPTO 2025. LNCS, vol. 16004, pp. 293–334. Springer (2025) 5
25. Kidmose, A.B., Tiessen, T.: A formal analysis of boomerang probabilities. IACR Trans. Symmetric Cryptol. **2022**(1), 88–109 (2022) 24
26. Kim, D., Kwon, D., Song, J.: Efficient computation of boomerang connection probability for arx-based block ciphers with application to SPECK and LEA. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **103-A**(4), 677–685 (2020) 3
27. Klimov, A., Shamir, A.: A new class of invertible mappings. In: CHES 2002. LNCS, vol. 2523, pp. 470–483. Springer (2002) 2
28. Leurent, G.: Analysis of differential attacks in ARX constructions. In: ASIACRYPT 2012. LNCS, vol. 7658, pp. 226–243. Springer (2012) 3, 12
29. Leurent, G.: Improved differential-linear cryptanalysis of 7-round chaskey with partitioning. In: EUROCRYPT 2016. LNCS, vol. 9665, pp. 344–371. Springer (2016) 26
30. Lipmaa, H., Moriai, S.: Efficient algorithms for computing differential properties of addition. In: FSE 2001. LNCS, vol. 2355, pp. 336–350. Springer (2001) 2, 15
31. Lipmaa, H., Wallén, J., Dumas, P.: On the additive differential probability of exclusive-or. In: FSE 2004. LNCS, vol. 3017, pp. 317–331. Springer (2004) 12
32. Liu, F., Sarkar, S., Meier, W., Isobe, T.: The inverse of χ and its applications to rasta-like ciphers. J. Cryptol. **35**(4), 28 (2022) 3, 4, 28
33. Machado, A.W.: Differential probability of modular addition with a constant operand. IACR Cryptol. ePrint Arch. p. 52 (2001) 3
34. Mealy, G.H.: A method for synthesizing sequential circuits. The Bell System Technical Journal **34**(5), 1045–1079 (1955) 1
35. Miyano, H.: Addend dependency of differential/linear probability of addition. IEICE Transactions on Fundamentals **E81-A**(1), 106–109 (1998) 17
36. Mouha, N., Mennink, B., Herrewege, A.V., Watanabe, D., Preneel, B., Verbauwhede, I.: Chaskey: An efficient MAC algorithm for 32-bit microcontrollers. In: SAC 2014. LNCS, vol. 8781, pp. 306–323. Springer (2014) 26
37. Mouha, N., Velichkov, V., Cannière, C.D., Preneel, B.: The differential analysis of s-functions. In: SAC 2010. LNCS, vol. 6544, pp. 36–56. Springer (2010) 2, 3, 7, 12, 16
38. Niu, Z.: The study of modulo 2^n. IACR Cryptol. ePrint Arch. p. 56 (2021) 3

39. Niu, Z., Sun, S., Liu, Y., Li, C.: Rotational differential-linear distinguishers of ARX ciphers with arbitrary output linear masks. In: CRYPTO 2022. LNCS, vol. 13507, pp. 3–32. Springer (2022) 3
40. Nyberg, K., Wallén, J.: Improved linear distinguishers for SNOW 2.0. In: FSE 2006. LNCS, vol. 4047, pp. 144–162. Springer (2006) 12
41. Schulte-Geers, E.: On ccz-equivalence of addition mod 2 n. Des. Codes Cryptogr. **66**(1-3), 111–127 (2013) 2
42. Tiessen, T.: Polytopic cryptanalysis. In: EUROCRYPT 2016. LNCS, vol. 9665, pp. 214–239. Springer (2016) 24
43. Wallén, J.: Linear approximations of addition modulo 2^n. In: FSE 2003. LNCS, vol. 2887, pp. 261–273. Springer (2003) 2, 12, 15
44. Wang, L., Song, L., Wu, B., Rahman, M., Isobe, T.: Revisiting the boomerang attack from a perspective of 3-differential. IEEE Trans. Inf. Theory **70**(7), 5343–5357 (2024) 4, 15, 24, 25

A Simultaneous column-monomialization of matrices

This appendix provides an efficient algorithm to find a basis that simultaneously column-monomializes a set of matrices, or to show that this is impossible. We still assume that $S_1 = S_2 = \cdots = S_n = S$. For every column-monomial matrix M, there exists a function $f\colon S \to S$ so that

$$M = D\,T^f\,, \tag{5}$$

where D is a diagonal matrix and T^f is the transition matrix of f. If M is of full rank, then T^f is a permutation matrix and hence M is a monomial matrix. However, as illustrated by Example 5, the matrices $T^{F_i}(\,\cdot\,;u,v)$ are generally not of full rank.

Definition 5. *A set of square matrices $\{M_1,\dots,M_m\}$ is simultaneously column-monomializable if there exists a change-of-basis matrix $\mathscr{B}$, diagonal matrices D_i and functions $f_i\colon S\to S$, such that $\mathscr{B}M_i\mathscr{B}^{-1} = D_iT^{f_i}$ for all i in $\{1,\dots,m\}$.*

A.1 Algorithm for simultaneous monomialization

If the matrices are simultaneously monomializable, like in Example 5, this problem reduces to simultaneous diagonalization. Indeed, since permutations have finite order, in this case there exists an integer $r > 0$ so that the matrices $M_1^r,\dots,M_m^r$ are simultaneously diagonalizable. A suitable change of basis can be found with high probability by choosing random $\alpha_1,\dots,\alpha_m$ from $\mathbb{K}$ and diagonalizing the matrix

$$\alpha_1M_1^r + \alpha_2M_2^r + \cdots + \alpha_mM_m^r\,.$$

However, this strategy does not work for simultaneous column-monomialization.

A.2 Algorithm for simultaneous column-monomialization

A first observation is that if a single basis vector is found, then multiplication by the matrices $M_1,\dots,M_m$ yields additional basis vectors. Indeed, if $\{b_1,b_2,\dots\}$ is a basis that simultaneously monomializes $M_1,\dots,M_m$, then

$$M_i\,b_j = \lambda_{i,j}\,b_{f_i(j)}\,.$$

Hence, the basis vector b_j yields basis vectors $b_{f_i(j)}, b_{f_i^2(j)},\dots$ for all i in $\{1,\dots,m\}$.

An initial set of basis vectors can be obtained by repeatedly computing intersections of the eigenspaces of $M_1^r,\dots,M_m^r$ corresponding to nonzero eigenvalues, where r is an integer so that $f_i^{2r} = f_i^r$. Concretely, one can choose $r = \operatorname{lcm}(1,2,\dots,N)$, where N is the number of rows and columns of the matrices M_i. Every intersection of dimension one yields a basis vector. Indeed, if the matrices M_i^r with $i \in I$ have a unique (up to scale) common eigenvector,

Algorithm 2 If it exists, find a basis $b_1, b_2, \ldots$ that simultaneously column-monomializes matrices $M_1, \ldots, M_m$ in $\mathbb{K}^{N \times N}$.

▷ *Compute sufficiently high powers of the matrices $M_1, M_2, \ldots$.* ◁
$r = \operatorname{lcm}(1, 2, \ldots, N)$
for $i = 1, 2, \ldots, m$ **do**
    $A_i \leftarrow M_i^r$
▷ *Intersect the eigenspaces of $A_1, \ldots, A_m$ to find joint eigenvectors.* ◁
$B \leftarrow \{\}$
$\Lambda \leftarrow$ Eigenspaces of $A_1, \ldots, A_m$
while $\dim \operatorname{span} B < N$ and $|\Lambda| \geq 1$ **do**
    ▷ *One-dimensional vector spaces in Λ yield basis vectors.* ◁
    for $V \in \Lambda$ if $\dim V = 1$ **do**
        $U \leftarrow \{\text{normalized basis vector of } V\}$ ▷ *Normalize for scale.*
        $U' \leftarrow \emptyset$
        while $U' \neq U$ and $\dim \operatorname{span} U' < N$ **do**
            $U \leftarrow U'$
            for $i = 1, 2, \ldots, m$ **do**
                $U' \leftarrow U' \cup \{\text{normalize}(M_i \times u) \mid u \in U\}$
        $B \leftarrow B \cup U'$
    ▷ *Compute pairwise intersections between the spaces in Λ.* ◁
    $\Lambda = \{V \cap W \mid V, W \in \Lambda \text{ if } V \cap W \not\subseteq \operatorname{span} B\}$
▷ *If B is incorrect, then $M_1, \ldots, M_m$ cannot be simultaneously monomialized.* ◁
return B

then there is a unique basis vector b_j with $f_i^r(j) = j$ for all i in I. A complete description is given in Algorithm 2.

Algorithm 2 is not guaranteed to terminate. This happens when there is no unique basis relative to which the matrices M_i^r are of the form $D_i T^g$, with g a idempotent function in the sense that $g \circ g = g$. However, in practice, this issue only occurs for exceptional choices of the matrices $M_1, \ldots, M_r$. In particular, computing the r^{th} power of each matrix M_i requires $\log r = \log e^{\mathcal{O}(N)} = \mathcal{O}(N)$ operations. If the algorithm terminates, then the number of iterations of the outer loop is at most m. In practice, the number of vector spaces in the set Λ does not increase much, because most intersections are empty once the spaces are sufficiently low-dimensional.

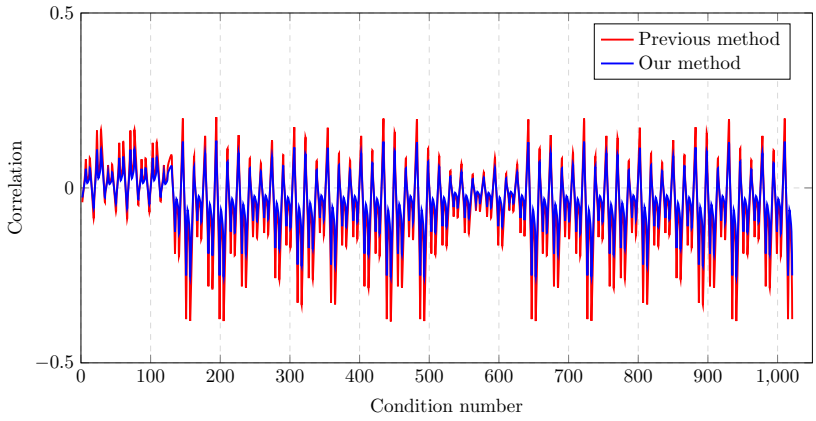
B Partitioning Conditions for LEA

The authors of [16] partition $\mathbb{F}_2^{7\times 32}$, into 2^{12} small spaces based on the following 12 equations parametrized by constants $v_1, v_2, \ldots, v_{12}$.

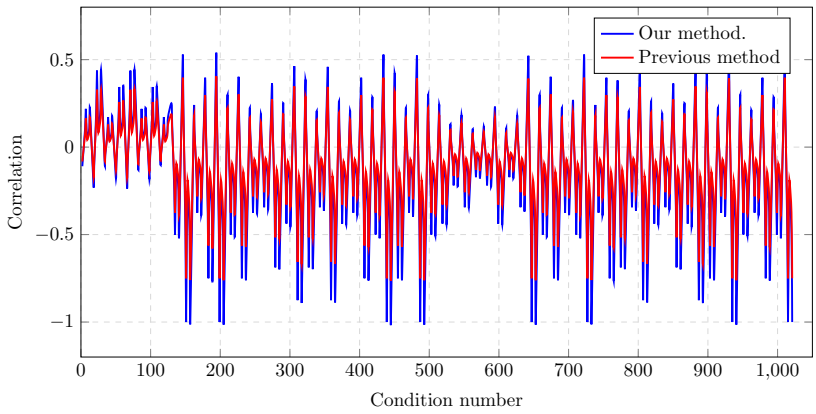
$$
\begin{aligned}
v_1 &= (y_0 + z_0 + k_0)[8]\,,\\
v_2 &= (y_0 + z_0 + k_0)[7]\,,\\
v_3 &= (y_0 + z_0 + k_0)[6]\\
v_4 &= \begin{cases} (y_1 + k_1)[8] + y_0[7] & \text{if } v_2 = 1\,,\\ (y_1 + k_1)[8] + y_0[6] & \text{if } v_2 = 0\,, \end{cases}\\
v_5 &= \begin{cases} (y_1 + k_1)[7] + y_0[6] & \text{if } v_3 = 1\,,\\ (y_1 + k_1)[7] + y_0[5] & \text{if } v_3 = 0\,, \end{cases},\\
v_6 &= \begin{cases} (y_2 + k_2)[8] + y_2[7] & \text{if } v_2 + v_5 = 1\,,\\ (y_2 + k_2)[8] + y_2[6] & \text{if } v_2 + v_5 = 0\,, \end{cases}\\
v_7 &= (y_0 + z_0 + k_0)[28]\,,\\
v_8 &= (y_0 + z_0 + k_0)[27]\,,\\
v_9 &= (y_0 + z_0 + k_0)[26]\,,\\
v_{10} &= \begin{cases} (y_1 + k_1)[28] + y_0[27] & \text{if } v_8 = 1\,,\\ (y_1 + k_1)[28] + y_0[26] & \text{if } v_8 = 0\,, \end{cases}\\
v_{11} &= \begin{cases} (y_1 + k_1)[27] + y_0[26] & \text{if } v_9 = 1\,,\\ (y_1 + k_1)[27] + y_0[25] & \text{if } v_9 = 0\,, \end{cases}\\
v_{12} &= (y_0 + z_0 + k_0)[25]\,.
\end{aligned}
$$

C Correlations for the LEA Round Function

If $(v_{12}, v_9) = (0, 0)$, then the estimated results in [16] are generally larger than the correct values. For $(v_{12}, v_9) = (0, 1)$, the estimated results in [16] are generally smaller than the correct values. The comparison is provided in Figure 9. When $(v_{12}, v_9) \notin \{(0, 0), (0, 1)\}$, the estimated correlations are consistent with our correct values.



(a) $(v_{12}, v_9) = (0, 0)$



(b) $(v_{12}, v_9) = (0, 1)$

Fig. 9: Correlations over the LEA round function. The horizontal axis is the integer value of $v_1 \| \cdots \| v_8 \| v_{10} \| v_{11} \in \mathbb{F}_2^{10}$ with v_1 as the most significant bit.

In [16], for the cipher $F \circ E$, the adversary selects a pair $(x, x \oplus \delta)$ for the subcipher E, and obtains the corresponding outputs $y = E(x)$ and $y' = E(x \oplus \delta)$, and then constructs a differential-linear distinguisher based on the linear approximation $\lambda^\top y + \lambda^\top y'$ with correlation r.

Then, the adversary queries N pairs $(x_i, x_i \oplus \delta)$, $1 \leq 0 \leq N-1$. For each ciphertext pair (m_i, m_i'), by guessing certain key bits, the adversary can determine the corresponding output value (w_i, w_i') for the subcipher F. For each (w_i, w_i'), depending on its value, we apply the corresponding conditional linear approximation from different partitions, which yield correlations of c_i and c_i' for $\lambda^\top y + \gamma^\top w_i$ and $\lambda^\top y' + \gamma' \cdot w_i'$, respectively. And for each pair, the adversary use $c_i \cdot c_i' \cdot r$ to calculate the LLR statistic for key recovery in the following

$$LLR = \frac{1}{2}\sum_{i=0}^{N-1} Ln(1-(rc_ic_i')^2) + \frac{1}{2}\sum_{i=0}^{N-1} \frac{Ln(1+(rc_ic_i'))}{Ln(1-(rc_ic_i'))}(-1)^{\gamma^\top w_i + \gamma' \cdot w_i'}.$$

According to [4], the LLR statistic follows normal distribution $\mathcal{N}(\mu_0, \sigma_0^2)$ and $\mathcal{N}(\mu_1, \sigma_1^2)$ for right guess and wrong guess, respectively, where

$$\mu_0 = -\mu_1 = \frac{1}{2}\sum_{i=0}^{N-1}(rc_ic_i')^2,\ \sigma_0^2 = \sigma_1^2 = \sum_{i=0}^{N-1}(rc_ic_i')^2.$$

Thus, we do the key-recovery attack by distinguishing between $\mathcal{N}(\mu_0, \sigma_0^2)$ and $\mathcal{N}(\mu_1, \sigma_1^2)$, and the larger value $\frac{u_0-u_1}{\sigma_0} = \sqrt{\sum_{i=0}^{N-1}(rc_ic_i')^2}$ indicates a higher distinguishing strength.

Taking into account all discrepancies in Figure 9, although the precision at (0,0) is lower than that calculated using the piling-up lemma in the previous method, all accuracy values under the (0,1) condition are significantly higher than those calculated using the piling-up lemma in the previous method. Moreover, the correlation under the (0,0) condition is far lower than that under the (0,1) condition. Furthermore, the discrepancy between the two methods is not particularly significant under the (0,0) condition, whereas it is highly significant under the (0,1) condition. If we use the exact correlation instead of the correlation computed via the piling-up lemma, the resulting $\sqrt{\sum_{i=0}^{N-1}(rc_ic_i')^2}$ will be larger. This implies that we can achieve the same effect with less data, or that both the filter strength and the success probability will be greater given the same amount of data.

D Matrices for Differential Cryptanalysis of Modular Addition

$$T^{F_i}(\,\cdot\,;\,q_{00,00},\,q^{0,0}) = \begin{bmatrix} \frac{3}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{3}{4} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{00,00},\,q^{0,1}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{00,00},\,q^{1,0}) = \begin{bmatrix} -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{00,00},\,q^{1,1}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & -\frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{00,01},\,q^{0,0}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{00,01},\,q^{0,1}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{00,01},\,q^{1,0}) = \begin{bmatrix} 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{00,01},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{00,10},\,q^{0,0}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{00,10},\,q^{0,1}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{00,10},\,q^{1,0}) = \begin{bmatrix} 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{00,10},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{00,11},\,q^{0,0}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{00,11},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{3}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & \frac{3}{4} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{00,11},\,q^{1,0}) = \begin{bmatrix} -\frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{00,11},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{01,00},\,q^{0,0}) = \begin{bmatrix} \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{01,00},\,q^{0,1}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{01,00},\,q^{1,0}) = \begin{bmatrix} \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{01,00},\,q^{1,1}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{01,01},\,q^{0,0}) = \begin{bmatrix} 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{01,01},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{01,01},\,q^{1,0}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{01,01},\,q^{1,1}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{01,10},\,q^{0,0}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{01,10},\,q^{0,1}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{01,10},\,q^{1,0}) = \begin{bmatrix} 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{01,10},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{01,11},\,q^{0,0}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{01,11},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{01,11},\,q^{1,0}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{01,11},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{4} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{10,00},\,q^{0,0}) = \begin{bmatrix} \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{10,00},\,q^{0,1}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{10,00},\,q^{1,0}) = \begin{bmatrix} \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{10,00},\,q^{1,1}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{10,01},\,q^{0,0}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{10,01},\,q^{0,1}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{10,01},\,q^{1,0}) = \begin{bmatrix} 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{10,01},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{10,10},\,q^{0,0}) = \begin{bmatrix} 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{10,10},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{10,10},\,q^{1,0}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{10,10},\,q^{1,1}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{10,11},\,q^{0,0}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{10,11},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & -\frac{1}{4} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{10,11},\,q^{1,0}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{10,11},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{4} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{11,00},\,q^{0,0}) = \begin{bmatrix} -\frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{11,00},\,q^{0,1}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & -\frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{11,00},\,q^{1,0}) = \begin{bmatrix} \frac{3}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{3}{4} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{11,00},\,q^{1,1}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{11,01},\,q^{0,0}) = \begin{bmatrix} 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{11,01},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{11,01},\,q^{1,0}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{11,01},\,q^{1,1}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{11,10},\,q^{0,0}) = \begin{bmatrix} 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{11,10},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{11,10},\,q^{1,0}) = \begin{bmatrix} 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{11,10},\,q^{1,1}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{11,11},\,q^{0,0}) = \begin{bmatrix} -\frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{11,11},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{1}{4} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{11,11},\,q^{1,0}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ \frac{1}{4} & 0 & -\frac{1}{4} & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{11,11},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{3}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & -\frac{3}{4} \end{bmatrix}$$

E Matrices for Differential Cryptanalysis of Modular Addition with A Constant

When the constant being added is zero:

$$T^{F_i}(\,\cdot\,;\,q_{0,0},\,q^{0,0}) = \begin{bmatrix} 1 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{0,0},\,q^{0,1}) = \begin{bmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{0,0},\,q^{1,0}) = \begin{bmatrix} 0 & 0 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{0,0},\,q^{1,1}) = \begin{bmatrix} 0 & \frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{0,1},\,q^{0,0}) = \begin{bmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{0,1},\,q^{0,1}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{0,1},\,q^{1,0}) = \begin{bmatrix} 0 & -\frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{0,1},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{1,0},\,q^{0,0}) = \begin{bmatrix} 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{1,0},\,q^{0,1}) = \begin{bmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{1,0},\,q^{1,0}) = \begin{bmatrix} 1 & 0 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{1,0},\,q^{1,1}) = \begin{bmatrix} 0 & \frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{1,1},\,q^{0,0}) = \begin{bmatrix} 0 & -\frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{1,1},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{1,1},\,q^{1,0}) = \begin{bmatrix} 0 & \frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix} \quad T^{F_i}(\,\cdot\,;\,q_{1,1},\,q^{1,1}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 0 \end{bmatrix}$$

When the constant being added is one:

$$T^{F_i}(\,\cdot\,;\,q_{0,0},\,q^{0,0}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} T^{F_i}(\,\cdot\,;\,q_{0,0},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{0,0},\,q^{1,0}) = \begin{bmatrix} -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} T^{F_i}(\,\cdot\,;\,q_{0,0},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{0,1},\,q^{0,0}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} T^{F_i}(\,\cdot\,;\,q_{0,1},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{0,1},\,q^{1,0}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix} T^{F_i}(\,\cdot\,;\,q_{0,1},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{1,0},\,q^{0,0}) = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} T^{F_i}(\,\cdot\,;\,q_{1,0},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & -\frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{1,0},\,q^{1,0}) = \begin{bmatrix} -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{2} & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} T^{F_i}(\,\cdot\,;\,q_{1,0},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{1,1},\,q^{0,0}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix} T^{F_i}(\,\cdot\,;\,q_{1,1},\,q^{0,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{2} & 0 & 0 & 0 \end{bmatrix}$$

$$T^{F_i}(\,\cdot\,;\,q_{1,1},\,q^{1,0}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} T^{F_i}(\,\cdot\,;\,q_{1,1},\,q^{1,1}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\frac{1}{2} & 0 & 0 & 0 \end{bmatrix}$$

F Matrices for Differential Cryptanalysis of Chi

$$T^{F_i}(\,\cdot\,;\,q_{0,0},\,q^{0,0}) =$$

$$\begin{bmatrix}
\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{0,0},\, q^{0,1}) =$

$$\begin{bmatrix}
0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{0,0},\, q^{1,0}) =$

$$\begin{bmatrix}
\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & -\frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{0,0},\, q^{1,1}) =$

$$\begin{bmatrix}
0 & \frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{0,1},\, q^{0,0}) =$

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{0,1},\, q^{0,1}) =$

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2}
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{0,1},\, q^{1,0}) =$

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{0,1},\, q^{1,1}) =$

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & -\frac{1}{2}
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{1,0},\, q^{0,0}) =$

$$\begin{bmatrix}
\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & -\frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{1,0},\, q^{0,1}) =$

$$\begin{bmatrix}
0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{1,0},\, q^{1,0}) =$

$$\begin{bmatrix}
\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{1,0},\, q^{1,1}) =$

$$\begin{bmatrix}
0 & \frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{1,1},\, q^{0,0}) =$

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{1,1},\, q^{0,1}) =$

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & -\frac{1}{2}
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;\, q_{1,1},\, q^{1,0}) =$

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0
\end{bmatrix}$$

$T^{F_i}(\,\cdot\,;q_{1,1},q^{1,1}) =$

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2}
\end{bmatrix}$$