Efficient and Extensive Search for Precise Linear Approximations with High Correlations of Full SNOW-V

Zhaocun Zhou[1,2*], Dengguo Feng[1,3] and Bin Zhang[1]

[1*]TCA Laboratory, SKLCS, Institute of Software, Chinese Academy of Science, Beijing, 100190, China.
[2]University of Chinese Academy of Science, Beijing, 100190, China.
[3]State Key Laboratory of Computer Science, ISCAS, Beijing, 100190, China.

*Corresponding author(s). E-mail(s): zhouzhaocun@126.com;
Contributing authors: martin_zhangbin@hotmail.com;

Abstract

SNOW-V is a stream cipher recently designed for 5G communication system. In this paper, we propose two efficient algorithms to evaluate the precise correlation of SNOW-V's two main nonlinear components with linear hull effects fully considered. Based on these algorithms, we could efficiently and extensively search much more linear masks than before. The ideas of these algorithms can be generalized to other similar nonlinear components in symmetric cipher. We apply our algorithms to full SNOW-V to search different types of linear approximations with high correlations. Our results depict more linear approximations with higher correlations than those proposed for full SNOW-V and SNOW-V$_{\boxplus_{32},\boxplus_8}$ recently. The best linear approximation we found has absolute correlation $\mathbf{2^{-47.567}}$. There are at least 8, 135 and 1092 linear approximations with absolute correlation greater than $\mathbf{2^{-47.851}}$, $\mathbf{2^{-49}}$ and $\mathbf{2^{-50}}$ respectively, which would derive a fast correlation attack with time/memory/data complexities $\mathbf{2^{240.86}}$, $\mathbf{2^{240.37}}$ and $\mathbf{2^{236.87}}$. It is better than all the previous results of fast correlation attack against full SNOW-V. Moreover, we propose some properties for linear trails

with 3 active S-boxes, which give a theoretical explanation that automatic search method lacks of. Our work provides a more comprehensive description for the linear approximation properties of full SNOW-V.

Keywords: Fast Correlation Attack, SNOW-V, Linear Approximation, Linear Hull, Depth-first, GPLFM

1 Introduction

Stream ciphers based on linear feedback shift register (LFSR) form an important class of stream cipher system. The history of these ciphers can be traced back to decades ago. The earlier ones commonly employ bit-wise LFSRs and memoryless nonlinear filter function, e.g., LILI-128 [1]. However, this type of stream cipher was soon found vulnerable to correlation attack [2], algebraic attack [3], etc. To enhance the resistance against those attacks, most of the modern stream ciphers exploit a finite state automata(FSM), which usually consists of several registers updated by nonlinear function.

A typical representative of word-LFSR based modern stream ciphers is the SNOW family. So far the SNOW family has 4 members including SNOW 1.0 [4], SNOW 2.0 [5], SNOW 3G [6] and SNOW-V [7]. They are mainly designed for fast implementation in software and widely used in communication system. SNOW 1.0 was submitted in the European project NESSIE. However, a guess-and-determine (GD) attack and a linear distinguishing attack are soon discovered [8]. Thereby, an enhanced version SNOW 2.0 is proposed. SNOW 3G is an improved version of SNOW 2.0 modified by the ETSI Security Algorithm Group of Experts(SAGE). A suit of 3GPP Confidentiality and Integrity Algorithms for UMTS and LTE networks named UEA2 & UIA2 employs SNOW 3G as a core cryptographic algorithm.

SNOW-V is a new family member proposed recently for the next generation 5G communication system. It shares a very similar structure with its predecessors SNOW 1.0/2.0/3G. Its first version was proposed on November 29, 2018, and a more developed version equipped with a byte-wise transposition σ was posted later [7]. The additional linear transformation σ makes SNOW-V much more stronger and hard to clarify its linear approximation property, as σ makes propagation of linear masks more complicated and brings non-negligible linear hull effect. Therefore, the linear approximations proposed early are for those simplified variants SNOW-V. The common variants are derived by replacing some of 32-bit adders with 8-bit version such as the reduced variant SNOW-V$_{\boxplus_{32},\boxplus_8}$ in [9] and the reduced variant in the specification [7], which simplifies the evaluation of linear approximations of 2 32-bit adders connected by transposition σ in SNOW-V. Another reduced variant is SNOW-V$_{\oplus}$ in which all 32-bit adders are replaced with XOR. These simplifications are adopted in the security evaluation based on linear approximations such as fast correlation attacks published at FSE 2020 [7] and FSE 2021 [9], and distinguishing attack

proposed in [10]. In addition, a guess-and-determine attack for SNOW-V is also presented in [10]. A distinguishing and key recovery attack on the reduced-round SNOW-V is proposed in [11]. Recently, an automatic linear trails search method by solving SMT/SAT model is proposed for full SNOW-V [12, 13]. The solutions of the model indicate much better linear trails with only 3 active S-boxes. Furthermore, 2 linear approximations with much higher correlations is given by exhaustively computing and summing up the correlations of linear trails to approximate linear hull effects.

Cryptanalysis approaches based on linear approximations of the FSM play an important role for these stream ciphers. Besides the linear distinguishing attack against SNOW 1.0 as mentioned above, the linear distinguish attack against SNOW 2.0 in [8] and SNOW 3G in [14], as well as the fast correlation attack(FCA) against SNOW 2.0 in [15] all belong to this class. FCA was pioneered by Meier and Staffelbach in 1989 [16], and later developed by [17, 18], etc. The initial state recovery problem of LFSR is transformed into a decoding problem in FCA. The linear part of the stream cipher is often treated as a linear code, while the nonlinear part is treated as noises stemming from a symmetric channel. For more related works, we refer to [19–23], etc. Some progresses in improving the details of the decoding algorithm also made in recent years [15, 24]. To perform anyone of these attacks, linear approximations with high correlation of the cipher must be found firstly.

Our Contributions

In this paper, we propose two algorithms to efficiently evaluate linear approximations of the FSM of SNOW-V with linear hull effect taken into consideration. Considering the nonlinear component that is composed of S-boxes substitution and modulo 2^{32} addition, a heuristic depth-first prune-and-search algorithm is proposed to find biased linear approximations, which exploits the empirical property that the correlation is expected to decrease when the number of active S-boxes increases. This technique drastically speeds up the search procedure, which allows us to evaluate much more potential linear masks than ever before. As for the nonlinear component 2 modulo 2^{32} additions connected via byte-wise transposition σ, which also has non-negligible linear hull effect, another algorithm is developed to evaluate its correlation of linear approximation. This algorithm allows us to compute linear approximations of the FSM of SNOW-V without any simplifications. Applying these techniques to SNOW-V, we could efficiently search precise linear approximations with higher correlations with 3, 12 or more active S-boxes in the linear trails. The best linear approximation we found has absolute correlation $2^{-47.567}$. Finally, 8, 135 and 1092 linear approximations with absolute correlation greater than $2^{-47.851}$, 2^{-49} and 2^{-50} are found out. Consequently, a fast correlation attack against full SNOW-V with time/memory/data complexity $2^{240.86}$, $2^{240.37}$ and $2^{236.87}$ is derived by these linear approximations. Our result is better than the previous results for full (or simplified) SNOW-V [9, 12, 13], and also provides a more comprehensive description for the linear approximation properties of full

Table 1 Comparison of the attacks against SNOW-V and its variants

Attack	Cipher	Time	Data	Memory	Reference
GD	full SNOW-V	2^{378}	8		[10]
Distinguishing	SNOW-V$_{\oplus}$	2^{303}	2^{303}		[10]
Fast correlation	SNOW-V$_{\boxplus_{32},\boxplus_8}$	2^{377}	2^{254}	2^{363}	[9]
Fast correlation	full SNOW-V	$2^{246.53}$	$2^{237.5}$	$2^{238.77}$	[12, 13]
Fast correlation	full SNOW-V	$2^{240.86}$	$2^{236.87}$	$2^{240.37}$	this paper

SNOW-V. The theoretical comparison with the automatic search method by solving SMT/SAT model is as follows.

Firstly, since our theoretical algorithms are efficient and also take linear hull effect into consideration, we could search much more potential good linear approximations and compute more precise correlations. Despite that automatic search method has advantage in finding biased linear trails one problem is that current SMT/SAT model can not describe the precise linear hull effect. In order to approximate linear hull effects, one has to solve the model for each indeterminate inner linear masks and sum up all the correlations of the linear trails. It is impractical when there are a lot of potential good indeterminate linear masks. The other problem is that it only provides very few solutions in one time solving, e.g, for the STP solver utilized in [12, 13], it outputs 1 solution by the COUNTEREXAMPLE instruction. Thereby, one has to exhaustively build and solve the SMT/SAT model for every outer masks to obtain more linear approximations with high correlations. It is also impractical when there are a lot of potential good outer linear masks in a linear approximation.

Secondly, we also propose some theoretical properties which are lacking in automatic search method for the linear trails with 3 active S-boxes. These properties not only help us in searching linear approximations, but also provide a theoretical description for the number, the patterns and the correlations of those linear trails. It also explains why the SMT/SAT will output these trails.

We expect this work could provide a more comprehensive description for the linear approximation properties of SNOW-V. Table 1 lists the existing attacks applied to SNOW-V or its simplified variants, and presents a comparison of our attack on SNOW-V with previous ones[1].

Outline

The rest of the paper is organized as follows. Section 2 lists some notations and gives a brief introduction of SNOW-V. Section 3 describes how to construct a three round linear approximation of SNOW-V. Section 4 proposes two algorithms to efficient search or compute correlations of the noises of main nonlinear components, and also elaborates the details of these two algorithms.

[1] As the complexity of correlation attacks for reduced variants given in [7] is very high, the result is not listed here.

Searching linear approximations for SNOW-V is given in Section 5. Finally, some conclusions are given in Section 7.

2 Preliminary

2.1 Notations and Definitions

Some notations are introduced for convenience.

- Given a n-bit vector $a = (a_{n-1}, \cdots, a_0)$, a_{n-1} denotes the most significant bit. When $b = (b_{4n-1}, \cdots, b_0)$ is a $4n$-byte vector, we use $b_{i,:} = (b_{4i+3}, \cdots, b_{4i})(0 \leq i < n)$ to denote its i-th column. When $y_{i,:}$ is treated as a 32-bit word, b_{4i+3} is the most significant byte.
- Given 2 n-bit vector $a = (a_{n-1}, \cdots, a_0)$, $b = (b_{n-1}, \cdots, b_0)$, if $a_{n-1} = \cdots = a_i = b_{n-1} = \cdots = b_i = 0$ and $a_{i-1} = b_{i-1} = 1$, we say that a and b have the same $n - i$ bits zero suffix, and denote it by $a^{sf} = b^{sf}$. Similarly, when a and b are all n-byte vectors, we could define the byte zero suffix and denote it by $a^{bsf} = b^{bsf}$. For convenience, let $End(a) = i - 1$ denote the index of the last nonzero byte.
- Let $\boxplus_{32}$ and $\boxplus_8$ denote the modulo 2^{32} and 2^8 additions respectively. Let $\boxplus$ denote the 16-byte addition in SNOW-V, which includes 4 modulo 2^{32} additions. Let $\oplus$ denote the bit-wise XOR operation.
- Given two n-bit vectors a and b, their inner product is defined as $a \cdot b = \oplus_{i=0}^{n-1} x_i y_i$.
- A n-byte(or $8n$-bit) linear mask is a binary vector, which will be denoted by a character in uppercase in the future, e.g., Γ or U. Let $\overline{\Gamma}$ denote the byte-wise pattern of Γ, which maps the zero byte of $\Gamma_{i,:}$ to 0 and nonzero byte to symbol $*$. For example, the byte-wise pattern of $(0\text{x}0\text{a}, 0\text{x}00, \cdots, 0\text{x}00)$ is $(*, 0, \cdots, 0)$.
- Let $W(\Gamma)$ and $W_b(\Gamma)$ denote the bit and byte Hamming weight of Γ respectively. The byte Hamming weight of 4 columns $(W_b(\Gamma_{0,:}), W_b(\Gamma_{1,:}), W_b(\Gamma_{2,:}), W_b(\Gamma_{3,:}))$ is denoted by $W_b(\Gamma_{I,:})$.
- The propagation from input mask to output mask is denoted by $\rightarrow$, e.g., $\Gamma \rightarrow \Lambda$.
- For a matrix M, we use $M^{(i,j)}$ to denote its element in the i-th row and j-th column, and M^t to denote its transposition.

Now we introduce the definition of linear approximation. Let $F : \mathbb{F}_2^m \rightarrow \mathbb{F}_2^n$ be a vectorial Boolean function. A linear approximation of F with m-bit input mask U and n-bit output mask pair V can be represented by

$$U \cdot x \oplus V \cdot F(x).$$

The correlation of linear approximation is used to measure the bias of a linear approximation.

$$c_F(U, V) = 2^{-m}(\{x : U \cdot x \oplus V \cdot F(x) = 0\} - \{x : U \cdot x \oplus V \cdot F(x) = 1\}).$$

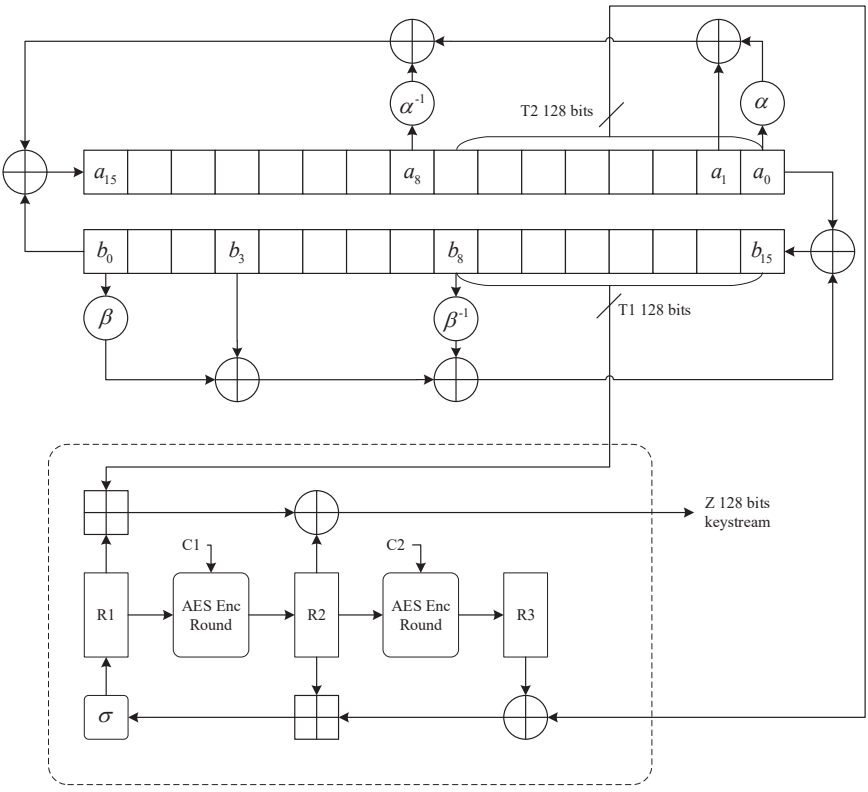

Fig. 1 The key stream generation phase of SNOW-V Algorithm

Relative entropy(also called Kullback–Leibler divergence) is used to measure the distance between two probability distributions.

Definition 1 Let p and q be probability density functions of two discrete probability distributions P and Q, their relative entropy(or Kullback-Leibler divergence) is defined by

$$D(p \parallel q) = \sum_{z} p(z) \log \frac{p(z)}{q(z)}.$$

2.2 A Brief Description of SNOW-V

SNOW-V shares a very similar structure with its predecessor SNOW 3G. The main difference is that the FSM of SNOW-V exploits larger registers updated by AES round function. Moreover, the latest version of SNOW-V is equipped with a byte-wise transposition σ. The overall schematic of the SNOW-V algorithm is depicted in Figure 1.

SNOW-V is composed by two layers. The linear part consists of two 16-cell LFSRs named LFSR-A and LFSR-B whose generator polynomials are

respectively

$$g^A(x) = x^{16} + x^{15} + x^{12} + x^{11} + x^8 + x^3 + x^2 + x + 1 \in \mathbb{F}_2[x],$$
$$g^B(x) = x^{16} + x^{15} + x^{14} + x^{11} + x^8 + x^6 + x^5 + x + 1 \in \mathbb{F}_2[x].$$

Let $(a_{t+15}, \cdots, a_{t+0})$ and $(b_{t+15}, \cdots, b_{t+0})$ denote their states at clock t respectively, the updates of the two LFSRs can be represented as follows.

$$a_{t+16} = b_t \oplus \alpha a_t \oplus a_{t+1} \oplus \alpha^{-1} a_{t+8},$$
$$b_{t+16} = a_t \oplus \beta b_t \oplus a_{t+3} \oplus \beta^{-1} a_{t+8},$$

where α and β are roots of $g^A(x)$ and $g^B(x)$ respectively.

The nonlinear layer has three 16-byte registers $R1, R2, R3$, which are updated by AES encryption round function.

$$R1_{t+1} = \sigma((T2_t \oplus R3_t) \boxplus R2_t),$$
$$R2_{t+1} = AES^R(R1_t),$$
$$R3_{t+1} = AES^R(R2_t),$$

where σ denotes the byte-wise transposition, and $T2_t = (a_{8t+7}, \cdots, a_{8t})$. The AES round function could be split into 4 operations, i.e., S-boxes substitution S, shift rows H, mixing columns M and round constant addition RC, $AES^R = RC \circ M \circ H \circ S$. Since RC only affects the sign of the correlation of linear approximation, it is neglected in the following cryptanalysis.

The key stream word Z_t is 16-byte, and generated by

$$Z_t = (T1_t \boxplus R1_t) \oplus R2_t,$$

where $T1_t = (b_{8t+15}, \cdots, b_{8t+8})$. For more details of SNOW-V algorithm, we refer to [7].

3 Linear Approximation of SNOW-V

In this section, we demonstrate how to construct binary linear approximations of SNOW-V, which only involve outputs of LFSR and key stream words. We need to seek the mask propagation property of FSM for several consecutive clocks. For SNOW-V, eliminating the register variables $R1$, $R2$ and $R3$, a binary linear approximation could be established with 3 consecutive LFSR outputs $T1_{t-1}$, $T1_t$, $T1_{t+1}$, $T2_{t-1}$, $T2_t$, $T2_{t+1}$ and key stream words Z_{t-1}, Z_t, Z_{t+1}. Linear approximations with more than 3 consecutive clocks could also be established with the similar technique. However, we have no evidence whether it is better than the case of 3 consecutive clocks when pursuing high correlation.

The three round linear approximations of SNOW-V are constructed as follows. Applying 3 binary linear masks Γ, Λ, Ψ to 3 consecutive key stream words Z_{t-1}, Z_t, Z_{t+1} respectively, we have

$$\begin{aligned}
\Gamma \cdot Z_{t-1} =& \Gamma \cdot S^{-1} \circ L^{-1}(R3_t) \oplus \Gamma \cdot \left(S^{-1} \circ L^{-1}(R2_t) \boxplus T1_{t-1}\right), \\
\Lambda \cdot Z_t =& \Lambda \cdot R2_t \oplus \Lambda \cdot (R1_t \boxplus T1_t), \\
\Psi \cdot Z_{t+1} =& \Psi \cdot L \circ S(R1_t) \oplus \Psi \cdot (\sigma(R2_t \boxplus (R3_t \oplus T2_t)) \boxplus T1_{t+1}),
\end{aligned} \tag{1}$$

where $L = M \circ H$. Linear masks Γ', Λ', Ψ and Λ^* are also applied to $T1_{t-1}, T1_t$, $T1_{t+1}$ and $T2_t$ respectively. These masks then propagate to other positions. For example, Λ propagates into nonlinear component, we may derive a linear approximation

$$\Lambda \cdot (Z_t \oplus R2_t) = \Lambda \cdot (T1_t \boxplus R1_t) = \Lambda \cdot (T1_t \boxplus S^{-1} \circ L^{-1}(y_t)) = \Lambda' \cdot T1_t \oplus \Psi \cdot y_t \oplus N1_t.$$

In order to take linear hull effect into account, a new variable $y_t = L \circ S(R1_t)$ is introduced here. The other linear approximations could be constructed similarly. Furthermore, in order to get a distinguisher for correlation attack, we require that [ref]

$$\Gamma' \cdot T1_{t-1} \oplus \Lambda' \cdot T1_t \oplus \Psi' \cdot T1_{t+1} \oplus \neq 0.$$

The details of linear mask propagation are depicted in Figure 2.

In summary, while linear hull effect is taken into consideration, the linear approximation of (1) may be divided into 3 parts, marked by colored dashed frame respectively in Figure 2. For convenience, they are called σ modular addition, S-boxes modular addition and duplicate S-boxes modular addition respectively. The noise variables $N1_t, N2_t, N3_t$ of 3 parts can be expressed as follows.

$$\begin{aligned}
N1_t =& \Lambda \cdot (S^{-1} \circ L^{-1}(y_t) \boxplus T1_t) \oplus \Lambda' \cdot T1_t \oplus \Psi \cdot y_t, \\
N2_t =& \Gamma \cdot S^{-1} \circ L^{-1}(R3_t) \oplus \Gamma \cdot (S^{-1} \circ L^{-1}(R2_t) \boxplus T1_{t-1}) \\
& \oplus \Lambda^* \cdot R3_t \oplus \Gamma^{(3)} \cdot R2_t \oplus \Gamma' \cdot T1_{t-1}, \\
N3_t =& \Psi \cdot (\sigma(R2_t \boxplus (R3_t \oplus T2_t)) \boxplus T1_{t+1}) \\
& \oplus \Lambda^{(3)} \cdot R2_t \oplus \Lambda^* \cdot R3_t \oplus \Lambda^* \cdot T2_t \oplus \Psi' \cdot T1_{t+1},
\end{aligned} \tag{2}$$

where $y_t = L \circ S(R1_t), \Gamma^{(3)} = \Lambda \oplus \Lambda^{(3)}$.

Consequently, the register variables $R1_t$, $R2_t$ and $R3_t$ will be eliminated by combining Equation (1), (2) together. More specifically, the relations between noises $N1_t$, $N2_t$, $N3_t$ and key stream words Z_{t-1}, Z_t, Z_{t+1} will be clear when

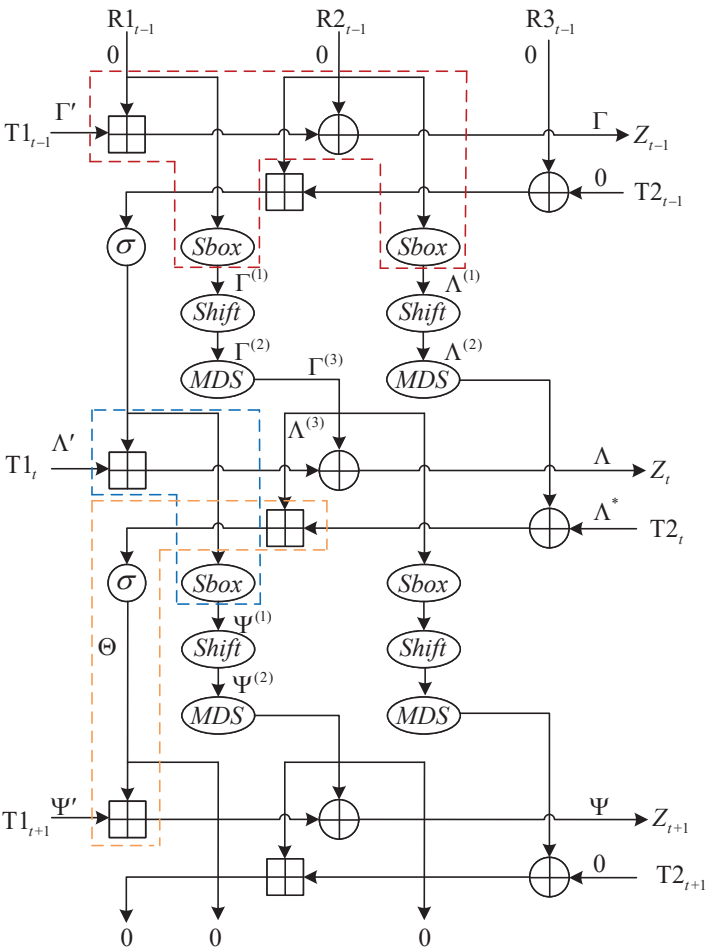

Fig. 2 The three round linear approximation of SNOW-V

the nonlinear parts are replaced by these noises. For example, since that

$$\begin{aligned}\Gamma \cdot Z_{t-1} =& \Gamma \cdot S^{-1} \circ L^{-1}(R3_t) \oplus \Gamma \cdot (S^{-1} \circ L^{-1}(R2_t) \boxplus T1_{t-1}),\\ N2_t =& \Gamma \cdot S^{-1} \circ L^{-1}(R3_t) \oplus \Gamma \cdot (S^{-1} \circ L^{-1}(R2_t) \boxplus T1_{t-1})\\ & \oplus \Lambda^* \cdot R3_t \oplus \Gamma^{(3)} \cdot R2_t \oplus \Gamma' \cdot T1_{t-1},\end{aligned}$$

it is derived that

$$\Gamma \cdot Z_{t-1} = N2_t \oplus \Lambda^* \cdot R3_t \oplus \Gamma^{(3)} \cdot R2_t \oplus \Gamma' \cdot T1_{t-1},$$

which reveals the relation between Z_{t-1} and $N2_t$. The other relations between Z_t, Z_{t+1} and $N1_t$, $N3_t$ could be derived similarly.

Once all relations are derived, the register variables $R1_t$, $R2_t$ and $R3_t$ would be eliminated. We will obtain a relation that only involves $N1_t$, $N2_t$, $N3_t$, Z_{t-1}, Z_t, Z_{t+1} and $T1_{t-1}$, $T1_t$, $T1_{t+1}$, $T2_t$. Finally, summing up all possible $\Gamma^{(3)}$ will construct a binary linear approximation of 3 consecutive clocks. Its correlation should be

$$c_{FSM} = \sum_{\Gamma^{(3)}} c(N1_t)c(N2_t)c(N3_t).$$

However, we may not run over all possible $\Gamma^{(3)}$, as our experiments reveal the following observation.

Observation 1. There exists one value of $\Gamma^{(3)}$ such that its corresponding linear path's correlation dominates the linear hull, and the other correlations are likely to be 0 or much smaller than the dominated one.

Notice that Γ' is derived from Λ^*, $\Gamma^{(3)}$ and Γ, while Λ' is derived from Λ and Ψ by pruning algorithm in the next section. And Γ', Λ^*, Γ, Λ, Ψ are all fixed values in a linear approximation of FSM. Intuitively, on one hand, current Γ' is not preferable for another $\Gamma^{(3)}$, which would reduce $c\,(N2_t)$. This case is verified by many examples of our experiments. On the other hand, as $\Lambda^{(3)} = \Gamma^{(3)} \oplus \Lambda$, $\Lambda^{(3)}$ changes when $\Gamma^{(3)}$ changes. As the masks Ψ, Λ^* and $\Lambda^{(3)}$ had been chosen with high $|c(N3_t)|$, another $\Lambda^{(3)}$ value would very likely reduce $|c(N3_t)|$, which is also observed in our experiments. Similar result as Observation 1 is also found in the cryptanalysis of SNOW-$V_{\boxplus_{32},\boxplus_8}$ in [9].

Meanwhile, since both noises $N2_t$ and $N3_t$ include variables $R2_t$ and $R3_t$, they are not independent theoretically. However, $N3_t$ and $R3_t \oplus T2_t$ are independent with each other. As for $R2_t$, we perform a small scale experiment to check the relative entropy between the union distribution and multiplication distribution. The result reveals strong independency, and see Appendix C. Therefore, it is reasonable to use their multiplication to approximate the overall correlation without much deviation.

Remark 1 The same (or similar) implicit independence assumption between two variables which actually have common inputs also appears in the cryptanalysis of SNOW-V [9, 12, 13] and ZUC [25]. So far it seems that there is no theoretical way to explain how the correlation varies if such dependence is taken into account. From these related works of SNOW-V and ZUC, it seems that people expect the correlation may be larger if such dependence is taken into account, even though no explanation or experiment is proposed. Precisely because of noticing this implicit assumption in previous related work, we perform a scaled experiment and check the assumption. At least it is an evidence that the assumption is reasonable.

4 Efficiently Computing the Correlations of Linear Approximations of the Noises

In this subsection, we discuss how to efficiently evaluate the correlations of noise variables $N1_t$, $N2_t$ and $N3_t$, which is the crucial part of our approximation. As both $N1_t$ and $N2_t$ contain S-boxes substitution and modular addition, while $N3_t$ includes 2 modular additions combined together by a byte-wise transposition, we propose two algorithms to efficiently compute their correlation respectively.

4.1 Efficient Search for Linear Approximations of $N1_t$ and $N2_t$ with high correlations.

Firstly, the linear approximation of S-boxes modular addition component in the blue dashed frame in Figure 2 could be divided into 4 independent columns with the following form

$$N = U \cdot (S^{-1}(y) \boxplus_{32} x) \oplus V \cdot x \oplus W \cdot y, \tag{3}$$

where U, V and W denote 32-bit linear masks and N is the noise variable.

Each column has 4 parallel S-boxes substitutions and a modulo 2^{32} addition. Since modulo 2^{32} addition is a type of T function, given binary masks U, V and W, it is quite efficient to evaluate $c(N)$ byte by byte with the rules of so called generalized pseudo-linear functions modulo 2^n (GPLFM). Obviously, if possible, storing the connection matrices will save time with the price of memory. For more details of such computation, we refer to [15, 26].

However, when U, V and W are not determined, in order to search for linear approximations with high correlation, more efficient algorithm is required to cover the mask space of U, V and W as large as possible. Since there are totally 2^{96} combinations of U, V and W, we propose a heuristic efficient search algorithm by observing connection matrices.

Let $(c_{i,0}, c_{i,1})^t = C_{u,v,w}(c_{i-1,0}, c_{i-1,1})^t$ denote a correlation transition via connection matrix $C_{u,v,w}$. Let $\mathcal{A}$ denote the set of all 2^{24} connection matrices $C_{u,v,w}$, $(u, v, w) \in \mathbb{F}_2^{8\times 3}$. Let $\mathcal{S}_0 \subset \mathcal{S}_1 \subset \mathcal{A}$ denote 2 subsets of $\mathcal{A}$, where

$$\begin{aligned}
\mathcal{S}_1 &= \{C_{u,v,w} \mid u \neq 0x00, v \neq 0x00, w \neq 0x00, u^{sf} = w^{sf}\},\\
\mathcal{S}_0 &= \{C_{u,v,w} \mid C_{u,v,w} \in \mathcal{S}_1, (c_{i,0}, c_{i,1})^t = C_{u,v,w}(c_{i-1,0}, c_{i-1,1})^t,\\
&\qquad c_{i,0} + c_{i,1} = c_{i-1,0} + c_{i-1,1} = 0\}.
\end{aligned}$$

Their sizes are $\#S_1 \approx 2^{22.41}$ and $\#S_0 \approx 2^{21.63}$ respectively.

For the case when $c_{i-1,0}c_{i-1,1} \geq 0$, we have $|c_{i-1,0} + c_{i-1,1}| \leq |c_{i,0}, c_{i,1}|$, as the sum of elements in a column ≤ 1, $\forall C_{u,v,w} \in \mathcal{A}$. The equality holds only if $C_{u,v,w} = C_{0,0,0}$. For the case when $c_{i-1,0}c_{i-1,1} < 0$, we need some further discussions.

Without loss of generality, suppose $|c_{i-1,0}| \geq |c_{i-1,1}|$, we rewrite $(c_{i-1,0}, c_{i-1,1}) = c_{i-1,0}(1, \delta - 1)$, $\delta = 1 + c_{i-1,1}/c_{i-1,0}$. Thus we have $c_{i,0} + c_{i,1} = c_{i-1,0}(b + a\delta)$, where $b = C^{(0,0)}_{u,v,w} + C^{(1,0)}_{u,v,w} - C^{(0,1)}_{u,v,w} - C^{(1,1)}_{u,v,w}$ and $a = C^{(0,1)}_{u,v,w} + C^{(1,1)}_{u,v,w}$. Obviously, we have $|c_{i-1,0} + c_{i-1,1}| \leq |c_{i,0}, c_{i,1}|$, $\forall C_{u,v,w} \in \mathcal{S}_0 \cup (\mathcal{A} \backslash \mathcal{S}_1)$.

For those matrices in $\mathcal{S}_1 \backslash \mathcal{S}_0$, we exhaustively check all the values of b and a. Most of them are very small. The absolute value upper bounds of b and a are respectively around 0.195 and 0.099. Thereby, we also have $|c_{i-1,0} + c_{i-1,1}| \leq |c_{i,0}, c_{i,1}|$ except that when $0 < \delta < 0.216$, $b = 0.195$, $a = 0.099$ or $0 < \delta < 0.177$, $b = 0.195$, $a = -0.099$ for very few connection matrices.

Thereby, we expected that the overall correlation almost decreases when the number of active S-boxes increases. We propose a heuristic depth-first prune-and-search algorithm to search linear approximations with high correlations, i.e., Algorithm 1. Particularly, we have

Proposition 1 *When the bit zero suffixes of masks* $(V_0, \cdots, V_i)$ *and* $(U_0, \cdots, U_i)$ *are not the same, i.e.,* $C_{U_i,V_i,W_i} \in \mathcal{A} \backslash (\mathcal{S}_1 \cup \{C_{0,0,0}\})$*, indeterminate correlation vector* $(c_{i,0}, c_{i,1})$ *must satisfy* $c_{i,0} + c_{i,1} = 0$*.*

Since if the bit zero suffixes are not the same, any one of linear trails of modulo 2^{32} addition will have zero correlation, which means that the most significant bit-slice of linear mask pairs is not (1,1,1). In the other words, there are some random "hanging" bits making the noise uniform random. Therefore, when $c_{i,0} + c_{i,1} \neq 0$, $(V_0, \cdots, V_i)$ and $(U_0, \cdots, U_i)$ must have the same bit zero suffix. An indeterminate correlation $c_{i,0} + c_{i,1} \neq 0$ may be observed. Thereby, we prune current branch in Algorithm 1, when the indeterminate absolute correlation is smaller than the current maximal value c_{max} or $c_{i,0} = c_{i,1} = 0$. The initial value c_{ini} affects the performance of Algorithm 1. The efficiency of Algorithm 1 will decrease when c_{ini} is too small, while the linear masks with high correlation may be missed when c_{ini} is too large.

Line 7 of Algorithm 1 involves connection matrices. The number of connection matrices is determined by the cell size and n. In our case, the cell is 8-bit and the maximal carry value is 1, there are $2^{8\times 3}$ 2×2 connection matrices. Storing them would consume about 512 MB memory when the data is represented by *double* type in C program language on a 64-bit computer, which is the memory complexity of the algorithm.

As for time complexity, we firstly concern about the distribution of correlation $c(N)$. Unfortunately, the distribution is not normal by a χ^2 test. Comparing with normal distribution, the peak at $c(N) = 0$ is much higher, and the tails is also higher, even though we restrict to $\mathcal{S}_1$. Without the information of the distribution, we may only derive a qualitative analysis of complexity bounds to reflect how much effort saved.

Here we assume that $C_{U_3,V_3,W_3} \in \mathcal{S}_1$. Let $|c_{thr}|$ denote the threshold correlation for pruning branch, i.e., current maximal absolute correlation during

Algorithm 1 Searching linear masks of Equation 3 with high correlation

Require: Potential binary mask Space $\mathcal{U} \times \mathcal{V} \times \mathcal{W}$.
Ensure: The maximal correlation c_{max} and linear masks U, V, W.

Precompute 2^{24} 2×2 GPLFM connection matrices, store them in set $\mathcal{C}$;
Let $c_{max} \leftarrow c_{ini}$, $U_{max} \leftarrow 0$, $V_{max} \leftarrow 0, W_{max} \leftarrow 0$, $(c_{-1,0}, c_{-1,1}) \leftarrow (1, 0)$;
for all possible pairs $(U, W) \in \mathcal{U} \times \mathcal{W}$ s.t. $U^{bsf} = W^{bsf}$ **do**
    Let depth $i \leftarrow 0$, $j \leftarrow End(U)$,$(c_{-1,0}, c_{-1,1}) \leftarrow (1, 0)$;
    for $V_i \in \mathbb{F}_2^8$ s.t. $i \leq j$ and $V \in \mathcal{V}$ **do**
        if $i = j$ and $V^{sf} \neq U^{sf}$ **then**
            continue;
        end if
        Compute correlation vector $(c_{i,0}, c_{i,1})^t \leftarrow C_{U_i,V_i,W_i}(c_{i-1,0}, c_{i-1,1})^t$ by connection matrix $C_{U_i,V_i,W_i} \in \mathcal{C}$ corresponding to (U_i, V_i, W_i);
        if $i = j$ and $|c_{j,0} + c_{j,1}| \geq c_{max}$ **then**
            $c_{max} \leftarrow |c_{j,0} + c_{j,1}|$, $U_{max} \leftarrow U, V_{max} \leftarrow V, W_{max} \leftarrow W$;
        end if
        if $i < j$ **then**
            if $c_{i,0} = c_{i,1} = 0$ or $(c_{i,0} + c_{i,1} \neq 0$ and $|c_{i,0} + c_{i,1}| < c_{max})$ **then**
                continue;
            else
                $i \leftarrow i + 1$;
            end if
        end if
    end for
end for
Return $c_{max}, U_{max}, V_{max}, W_{max}$.

the loops. We are interesting about the probability that current path terminate at the i-th level if we only set the prune condition at the i-th byte. Assume the probability that the path terminates at the i-th level is p_i. Therefore, there are $p_i N_i$ masks will terminate at i-th level on average, where $N_i = 2^{24i} \# \mathcal{S}_1$ denotes the number masks maybe induce a nonzero indeterminate correlation at i-th level. Thus an upper bound of time complexity would be $N_3 - p_i N_i 2^{24(3-i-1)} \# \mathcal{S}_1$. Here the unit of the complexity of computing correlation one time.

Suppose that if a mask tuple $(U, V, W)_{\leq i} = ((U_0, \cdots, U_i), (V_0, \cdots, V_i), (W_0, \cdots, W_i))$ can be terminated at i-th level, then $(U, V, W)_{\leq i+1}$ is very likely to be terminated at $i + 1$-th level as long as $(U, V, W)_{\leq i}$ is a prefix of $(U, V, W)_{\leq i+1}$ and $C_{U_{i+1},V_{i+1},W_{i+1}} \in \mathcal{S}_1$. Now we could similarly estimate the average number of paths pruned at each level, when all prune conditions are

set. The paths terminated at 0th, 1st and 2nd level are respectively

$$\begin{aligned} Q_0 &= p_0 N_0, \\ Q_1 &= p_1 N_1 - Q_0 \#\mathcal{S}_1, \\ Q_2 &= p_2 N_2 - Q_1 \#\mathcal{S}_1 - Q_0 2^{24} \#\mathcal{S}_1. \end{aligned}$$

The 3rd level has remaining $Q_3 = (2^{24\times 3} - \sum_{i=0}^{2} Q_i 2^{24\times(2-i)})\#\mathcal{S}_1$ paths. Thus the time complexity can be approximated by $\sum_{i=0}^{3} Q_i$.

As for the case corresponding to the red dash frame in Figure 2, the linear approximation of the nonlinear component can still be divided into 4 columns represented by the following linear approximation

$$N = U \cdot ((S^{-1}(y) \boxplus_{32} x) \oplus S^{-1}(z)) \oplus V \cdot x \oplus W \cdot y \oplus E \cdot z. \tag{4}$$

Similar pruning tactics could also be applied. A difference is that U and E is required to have the same byte pattern, i.e., $U_i = 0$ if and only if $E_i = 0$. The specific algorithm is depicted in Appendix A.

4.2 Efficient Evaluation of the Correlation of $N3_t$.

Next, we deal with the nonlinear part in orange dashed frame. A binary linear approximation of this nonlinear operation would be

$$N = U \cdot (x \boxplus \sigma(y \boxplus z)) \oplus V \cdot x \oplus W \cdot y \oplus T \cdot z, \tag{5}$$

where U, V, W and T are all 16-byte linear masks.

Notice that transposition σ fuses 2 32-bit adders together, the carry information of $\sigma(y \boxplus z)$ firstly propagates in row direction, and the next $\boxplus$ makes carry information propagates in column direction. Motivated by this phenomenon, we could give an artful solution to evaluate the correlation of Equation (5). The overall schematic are depicted in Figure 3. In this subsection, we use uppercase letter and lowercase letter with a subscript to denote a byte and a bit respectively, e.g., the i-th byte of x is denoted by X_i, while the i-th bit of x is denoted by x_i. For convenience, let us introduce some notations to describe the complicated carry propagation schematic.

Let cr_i and cc_i, $i \in \{0, 1, \cdots, 127\}$, denote the carry bits for $y \boxplus z$ and $x \boxplus o$ respectively, then we have

$$cr_{i+1} = \begin{cases} y_i z_i \vee cr_i z_i \vee y_i cr_i & \text{if } i \notin \{0, 32, 64, 96\} \\ y_i z_i & \text{if } i \in \{0, 32, 64, 96\} \end{cases}, cr_0 = 0.$$

$$cc_{i+1} = \begin{cases} y_i o_i \vee cc_i o_i \vee y_i cc_i & \text{if } i \notin \{0, 32, 64, 96\} \\ y_i o_i & \text{if } i \in \{0, 32, 64, 96\} \end{cases}, cc_0 = 0.$$

where $o = \sigma(y \boxplus z)$ and $\vee$ denotes bit OR.

Therefore, $(cr_{8\sigma(i)+j}, cc_{8i+j})$ represents the carry vector of σ modular addition at j-th bit of i-th byte. Let $c_{8i+j,\delta}$ denote the correlation value at this position under the condition $cr_{8\sigma(i)+j}2 + cc_{8i+j} = \delta$. Thereby, column vector $\boldsymbol{c}_i = (c_{i,0}, c_{i,1}, c_{i,2}, c_{i,3})$ represents correlation propagation information from i-th bit to $(i+1)$-th bit, which is called correlation propagation vector for convenience. We could construct 16 4×4 connection matrices corresponding to bit slice value (u_i, v_i, w_i, t_i) of U, V, W, T similarly as previous, which would connect correlation propagation vector between i-th and $(i+1)$-th bit, i.e., $\boldsymbol{c}_{i+1} = A_{u_i 2^3 + v_i 2^2 + w_i 2 + t_i} \cdot \boldsymbol{c}_i$.

Since the 5 carries $cr_{8(i+1)}$, $cr_{8(i+1)+32}$, $cr_{8(i+1)+64}$, $cr_{8(i+1)+96}$ and $cc_{32(i+1)}$ are the carries from i-th column of $x \boxplus \sigma(y \boxplus z)$ to $(i+1)$-th column, We use two arrays M_0, M_1 of size 2^5 to store the correlation when the 5 carries take different values. Algorithm 2 indicates how to compute the correlation of $N3_t$.

Algorithm 2 Computing the correlation of Equation (5)

Require: 16-byte linear masks U, V, W, T.
Ensure: correlation value cor.

Precompute 16 4×4 connection matrices $A_0, \cdots, A_{15}$;
Initialize the first element $M_0[0] \leftarrow 1$, the other elements with 0;
for column index i from 0 to 3 **do**
    $M_1 \leftarrow \text{RowCarryCorrelation}(M_0, i)$, $M_0 \leftarrow M_1$;
end for
Return $cor \leftarrow \sum_{i=0}^{31} M_0[i]$.

Algorithm 3 RowCarryCorrelation

Require: Array M_0 corresponding to the $(i-1)$-th column and current column index i.
Ensure: Array M_1 corresponding to the i-th column.

Initialize temporary arrays $G_0 \leftarrow M_0$, G_1 and M_1 with 32 0s;
if $i = 0$ **then**
    let set $\mathcal{K} \leftarrow \{(0,0,0,0)\}$;
else
    $\mathcal{K} \leftarrow \{0,1\}^4$;
end if
for each $\boldsymbol{k} = (k_0, k_1, k_2, k_3) \in \mathcal{K}$ **do**
    for row index j from 0 to 3 **do**
        $G_1 \leftarrow \text{ByteCorrelation}(G_0, j, \boldsymbol{k})$, $G_0 \leftarrow G_1$;
    end for
    $M_1[0] \leftarrow M_1[0] + G_0[0], \cdots, M_1[31] \leftarrow M_1[31] + G_0[31]$;
end for
Return M_1.

Algorithm 4 ByteCorrelation

Require: Array G_0, current row index j and previous row carry vector $\boldsymbol{k}$.
Ensure: Array G_1.

Initialize temporary arrays G_1 and G_2 with 32 0s;
if $j = 0$ **then**
  $C_0 \leftarrow (G_0[\sum_{i=0}^{3} k_i 2^{4-i}] + G_0[\sum_{i=0}^{3} k_i 2^{4-i} + 1], 0, 0, 0)$;
  if $k_0 = 1$ **then**
    swap $C_0[0]$ and $C_0[2]$;
  end if
  for bit index l from 0 to 7 **do**
    $C_1 \leftarrow A_{u_l 2^3 + v_l 2^2 + w_l 2 + t_l} \cdot C_0$, $C_0 \leftarrow C_1$;
  end for
  $G_1[0] \leftarrow C_0[0], \cdots, G_1[3] \leftarrow C_0[3]$;
else
  for previous carry crp from 0 to $2^j - 1$ **do**
    $C_0 \leftarrow (G_1[2crp], G_1[2crp + 1], 0, 0)$;
    if $k_j = 1$ **then**
      swap $C_0[0]$ and $C_0[2]$, swap $C_0[1]$ and $C_0[3]$;
    end if
    for bit index l from 0 to 7 **do**
      $C_1 \leftarrow A_{u_l 2^3 + v_l 2^2 + w_l 2 + t_l} \cdot C_0$, $C_0 \leftarrow C_1$;
    end for
    $G_2[4crp] \leftarrow C_0[0], \cdots, G_2[4crp + 3] \leftarrow C_0[3]$;
  end for
  $G_1 \leftarrow G_2$;
end if
Return G_1.

The detailed basic idea of Algorithm 2 is as follows. We know that each byte of $X \boxplus \sigma(Y \boxplus Z)$ could be expressed as

$$X_i \boxplus_8 (Y_j \boxplus_8 Z_j \boxplus_8 cr_k) \boxplus_8 cc_l,$$

The row carry cr_k propagates in row direction, while the column carry cc_l propagates in column direction. For example, for $X_5 \boxplus_8 (Y_5 \boxplus_8 Z_5 \boxplus_8 cr_{40}) \boxplus_8 cc_{40}$, row carry cr_{40} is derived from $Y_4 \boxplus_8 Z_4$ in the left, and column carry cr_{40} is derived from $X_4 \boxplus_8 (Y_1 \boxplus_8 Z_1 \boxplus_8 cr_8)$ in the above.

For each value of $(cr_8, cr_{40}, cr_{72}, cr_{104}) \in \{0, 1\}^4$, we compute the correlation of the first column. Then for each input carries $(cr_8, cr_{40}, cr_{72}, cr_{104})$, we would iteratively compute the correlation of the second column for different output row carries $(cr_{16}, cr_{48}, cr_{80}, cr_{112})$. Finally, the sum of these rows is the output correlation of the second column for different output row carries $(cr_{16}, cr_{48}, cr_{80}, cr_{112})$, see Figure 4. This process can be performed column by column.

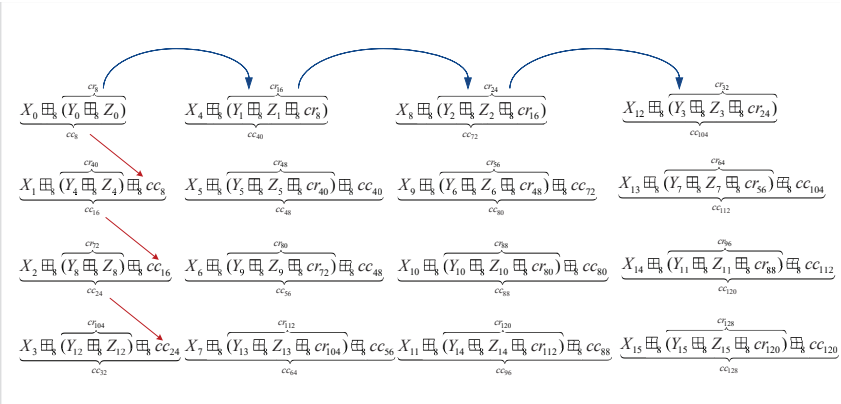

Fig. 3 Carry propagation for $X \boxplus \sigma(Y \boxplus Z)$

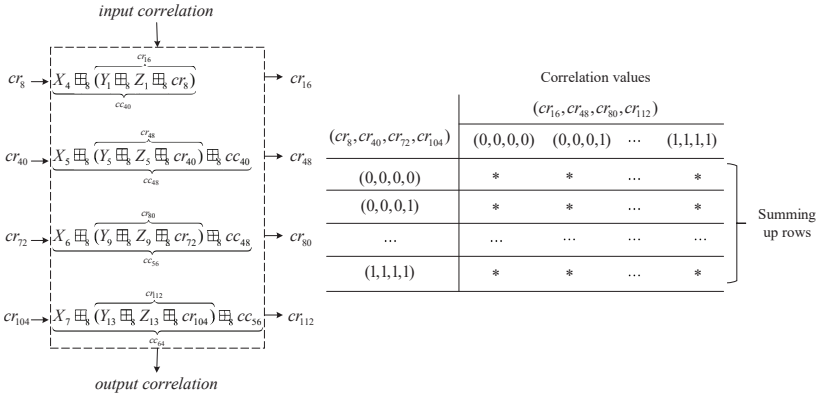

Fig. 4 Computing correlation from the first column to the second column

For each column, we compute the correlation byte by byte from the top to the bottom. When current byte is done, we store the results according to the next row carry. For example, we store the correlation of $X_4 \boxplus_8 (Y_1 \boxplus_8 Z_1 \boxplus_8 cr_8)$ according to the next row carry cr_{16}, when we arrived at the second byte $X_5 \boxplus_8 (Y_5 \boxplus_8 Z_5 \boxplus_8 cr_{40}) \boxplus_8 cc_{40}$. Similarly, current correlations are stored according to the different values of (cr_{16}, cr_{48}), when we arrived at the third byte $X_6 \boxplus_8 (Y_9 \boxplus_8 Z_9 \boxplus_8 cr_{72}) \boxplus_8 cc_{48}$. The idea is depicted in Figure 5. Some skills such as folding or swapping the correlation vector are needed when we are walking from current byte to the next byte. The reason is that the row carry propagates to the next column rather than the next byte. For example, cr_{16} is not added into $X_5 \boxplus_8 (Y_5 \boxplus_8 Z_5 \boxplus_8 cr_{40}) \boxplus_8 cc_{40}$. For each byte, correlation could be computed by connection matrices. Some experimental results of Algorithm 2 is presented in Appendix D.

As stated above, the 2 additions $\boxplus$ in σ modular addition are separated and performed by order, we could firstly compute a carry correlation array M_0

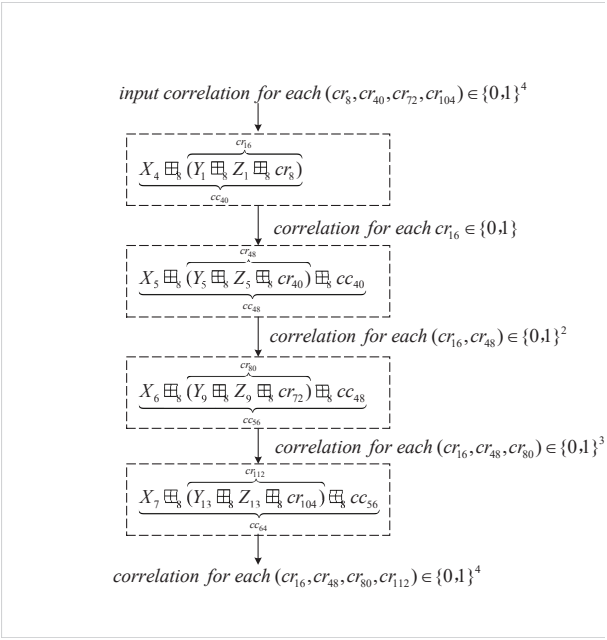

Fig. 5 Computing correlation byte by byte

of i-th column, which indicates the correlation when $\boldsymbol{c}_i^{col}$ appears. Then we iteratively compute the following columns, and see line 3 $\sim$ 4 in Algorithm 2.

Algorithm 3 and 4 show how to glue two adjacent columns. When evaluating M_0 of the $(i+1)$-th column, all possible $\boldsymbol{c}_i^{col}$ are searched, and all results are summed up, and see line 2 $\sim$ 7 in Algorithm 3. The output correlation value related to $cc_{32(i+1)} = 1$ is added to correlation related to $cc_{32(i+1)} = 0$, because that $32(i+1)$ is the index of the least significant bit of X's column, and see line 3 in Algorithm 4. The input carry at j-th bit of the i-th byte is $(cr_{8\sigma(i)+j}, cc_{8i+j})$, which has 4 possible values. Given the output carry $(cr_{8\sigma(i-4)+8}, cc_{8(i-4)+8})$, the first two components and last two components of correlation propagation vector, which is the input at 0-th bit of the i-th byte, are swapped when $cr_{8\sigma(i-4)+8} = 1$, and see line 4 and 11 in Algorithm 4.

Since there are 4 columns and 4 rows in total, the time complexity of Algorithm 2 is about 2^{11} 4×4 matrix multiplications. The memory complexity is about $32 \times 3 + 14 \times 4 = 160$ *double* numbers. To verify the validity of Algorithm 2, a scaled experiment is performed for which there are 2 columns by 2 rows with 2-bit cell size. Furthermore, as for the case of 4 columns by 4 rows and 8-bit cell size, we compute the empirical correlations of some examples by random samples of X, Y and Z. Our experiments show that their empirical correlations are close to the theoretical value. One of these examples is given in Appendix C. We also point out that the basic idea of Algorithm 2 may be applied to other similar cases, which only involve modular additions.

5 Extensive Search for Precise Linear Approximations of Full SNOW-V

In this subsection, we apply above algorithms to extensively search precise linear approximations of full SNOW-V. We focus on those linear trails with 3, 12 or even more active S-boxes. Notice that the linear trails proposed in [12, 13] for full SNOW-V have 3 active S-boxes, while the linear trails utilized in [9] for simplified SNOW-V have 12 active S-boxes.

5.1 Improving the Linear Approximations with 3 Active S-boxes

In [12, 13], a linear approximation with high correlation for full SNOW-V is proposed. It includes linear trails with only 3 active S-boxes. The columns propagation $\Lambda^{(2)}_{i,:} \to \Lambda^{*}_{i,:}$ and $\Psi^{(2)}_{i,:} \to \Psi_{i,:} (0 \leq i < 4)$ both have the following byte pattern

$$(0,0,0,*,0,0,0,0,0,0,0,0,0,0,0,0) \to (*,*,*,*,0,0,0,0,0,0,0,0,0,0,0,0).$$

They are obtained by constructing SMT/SAT models and solving by STP solver, and are the best linear trails known for SNOW-V so far.

This automatic search method has significant advantage to indicate linear trails with high correlations, especially in linearly approximating complicated FSM. However, some limitations should be noticed. Firstly, the FSM is divided into into several sub-steps by introducing sub-layer functions. The linear approximation for these sub-layer functions are treated as independent variables. Their correlations are multiplied together as the overall correlation, while the linear hull effects are omitted. Thereby, the solution of the current SMT/SAT model only reveals the linear trail with high correlation rather than the precise correlation of the outer linear approximation. Secondly, large-scale extensive search is impractical. Since a number of biased linear approximations will help to reduce the correlation attack complexity, we hope to obtain linear approximations with high correlations as many as possible. However, it is impractical to exhaustively search a large number of outer and indeterminate inner linear masks each by solving an automatic model.

Our algorithms are used to perform a more comprehensive search for precise linear approximations of which linear trails has only 3 active S-boxes. To reduce the search space, we first give some properties about those linear trails with 3 active S-boxes.

Proposition 2 *Suppose that* $\Lambda^{(3)}$, Λ^{*}, Ψ' *and* Ψ *all has one active column,* Λ^{*} *and* Ψ *has 4 active bytes in that column, then the only feasible* Λ^{*} *and* Ψ *that may induce nonzero correlation for* σ *modular addition has byte pattern* $(*,*,*,*,0,0,0,0,0,0,0,0,0,0,0,0)$*, i.e., the 3rd column is active.*

Proof Suppose the i-th column of Λ^* is active, since the correlation of σ modular addition could be expressed by

$$c_\sigma = \sum_{\Theta} c(\Psi, \Psi', \Theta) c(\sigma(\Theta), \Lambda^{(3)}, \Lambda^*),$$

the i-th column of $\Lambda^{(3)}$ is also active, or $c(\Theta, \Lambda^{(3)}, \Lambda^*)$ is 0. Similarly, we have that the j-th column of Ψ' and Ψ is active.

As there is a σ between $(y \boxplus z)$ and x, we conclude that the byte at the i-th row and j-th column Θ_{4j+i} must be nonzero and all the other bytes of Θ is zero. Otherwise, the correlation of this linear trail in linear hull $(\Lambda^{(3)}, \Lambda^*, \Psi', \Psi)$ must be zero. For example, suppose that there is another byte $\Theta_{4j+i'}, i' \neq i$ is nonzero, then $\Theta_{4j+i'}$ must introduce the variable about the byte at i'-column and j-th of $y \boxplus z$. Since y and z both are random, the $(4i' + j)$-th byte of $y \boxplus z$ is also random, and this variable only appears once, hence the correlation of current linear trail is 0.

Moreover, it is well known that the correlation $c(\Psi_{j,:}, \Psi'_{j,:}, \Theta_{j,:}) = 0$, when the bit zero suffixes of $\Psi_{j,:}$, $\Psi'_{j,:}$ and $\Theta_{j,:}$ are not the same [8]. As Ψ has 4 active bytes in the active column, thus we have $j = 3$. Similarly, we have $i = 3$. □

From the proof of Proposition 2, we have the following corollary.

Corollary 3 *Suppose the conditions are the same as Proposition 2, then the byte pattern of the indeterminate linear mask* Θ *is* $(*, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$.

We can even deduce the form of good $\Lambda^{(3)}$ and Ψ' when there are only 3 active S-boxes.

Proposition 4 *Suppose there are 3 active S-boxes in a linear trail of three-round of SNOW-V, let* $c(\Psi, \Psi', \Theta, \Lambda^{(3)}, \Lambda^*) = c(\Psi, \Psi', \Theta) c(\sigma(\Theta), \Lambda^{(3)}, \Lambda^*)$ *denote the correlation of linear trail passing through* σ *modular addition, then the upper bound of* $|c(\Psi, \Psi', \Theta, \Lambda^{(3)}, \Lambda^*)|$ *is* 2^{-36}*, and only one trail* $(\Psi, \Psi', \Theta, \Lambda^{(3)}, \Lambda^*)$ *satisfies the bound.*

Proof Since there are 3 active S-boxes in the linear trail, each of $\Gamma^{(2)}$, $\Lambda^{(2)}$ and $\Psi^{(2)}$ has 1 active byte in the 3rd column, i.e, the last column.

Firstly, we consider $c(\Psi_{3,:}, \Psi'_{3,:}, \Theta_{3,:})$. As there is a shift row operation when $\Psi^{(2)} \to \Psi^{(1)}$, the byte pattern of $\Psi^{(2)}$ must be $(0, 0, 0, *, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$. Otherwise, there is a nonzero byte not in $\Lambda^{(3)}_{3,:}$, which will deduce a correlation 0 of the overall linear trail. Thereby, the mask tuple $(\Psi_{3,:}, \Psi'_{3,:}, \Theta_{3,:})$ of modulo 2^{32} addition has the following form.

$$\begin{aligned}
\Psi_{3,:} &: \psi_{127} \cdots \psi_{120}, \psi_{119} \cdots \psi_{112}, \psi_{111} \cdots \psi_{104}, \psi_{103} \cdots \psi_{96}, \\
\Psi'_{3,:} &: \psi'_{127} \cdots \psi'_{120}, \psi'_{119} \cdots \psi'_{112}, \psi'_{111} \cdots \psi'_{104}, \psi'_{103} \cdots \psi'_{96}, \\
\Theta_{3,:} &: \theta_{127} \cdots \theta_{120}, 00000000, 00000000, 00000000.
\end{aligned}$$

According to state transition automata proposed in [8], the state transition is influenced by the Hamming weight $W(\psi_i, \psi'_i, \theta_i)$ of the bit-slice. As $W(\psi_i, \psi'_i, \theta_i) \neq$

3 when $104 \leq i < 120$, if there is a $W(\psi_i, \psi'_i, \theta_i) = 1, 105 \leq i < 120$, then $c(\Psi_{3,:}, \Psi'_{3,:}, \Theta_{3,:}) = 0$. Thus we have $(\psi_{119} \cdots \psi_{105}) = (\psi'_{119} \cdots \psi'_{105})$. As $(\psi_{103} \cdots \psi_{96}) \neq 0x00$, the optimal choice for high correlation is that $(\psi_{119} \cdots \psi_{104}) = (\psi'_{119} \cdots \psi'_{104})$, $(\psi_{103} \cdots \psi_{96}) = 0x80$ and $(\psi'_{103} \cdots \psi'_{96}) = 0x00$. The optimal choice for the most significant byte is that

$$(\psi_{127} \cdots \psi_{120}) = (\psi'_{127} \cdots \psi'_{120}) = (\theta_{127} \cdots \theta_{120}) = 0x01.$$

If this is the case, $|c(\Psi_{3,:}, \Psi'_{3,:}, \Theta_{3,:})| = 2^{-17}$. However, there is no such $\Psi^{(2)}_{3,:} \to \Psi_{3,:}$ satisfying above conditions. The only suboptimal choice is that

$$\begin{aligned}(\psi_{127} \cdots \psi_{120}) = (\psi'_{127} \cdots \psi'_{120}) &= 0x81\\ (\theta_{127} \cdots \theta_{120}) = 0xc1, (\psi_{103} \cdots \psi_{96}) &= 0x80\\ (\psi_{119} \cdots \psi_{104}) &= (\psi'_{119} \cdots \psi'_{104}).\end{aligned}$$

The correlation $|c(\Psi_{3,:}, \Psi'_{3,:}, \Theta_{3,:})| = 2^{-18}$. The only choice is that $\Psi^{(2)}_{3,:} = 0x000000b7$, $\Psi_{3,:} = 0x81ec5a80$.

Secondly, we consider $c(\sigma(\Theta), \Lambda^{(3)}, \Lambda^*)$. $\Lambda^{(2)}_{3,:}$ has 4 possible byte patterns $(*, 0, 0, 0)$, $(0, *, 0, 0)$, $(0, 0, *, 0)$ and $(0, 0, 0, *)$. The process of analysis is similar as above. The only suboptimal choice is also that

$$\begin{aligned}(\lambda^{(3)}_{127} \cdots \lambda^{(3)}_{120}) = (\lambda^*_{127} \cdots \lambda^*_{120}) &= 0x81\\ (\theta_{127} \cdots \theta_{120}) = 0xc1, (\lambda^*_{103} \cdots \lambda^*_{96}) &= 0x80\\ (\lambda^*_{119} \cdots \lambda^*_{104}) &= (\lambda^{(3)}_{119} \cdots \lambda^{(3)}_{104}).\end{aligned}$$

Therefore, the upper bound of $|c(\Psi, \Psi', \Theta, \Lambda^{(3)}, \Lambda^*)|$ is 2^{-36}. There is only one trail $(\Psi, \Psi', \Theta, \Lambda^{(3)}, \Lambda^*)$ satisfies the bound. □

.

Remark 2 Notice that the free mask Ψ' is not that free from Proposition 4. The best of those (Ψ, Ψ') are listed in Appendix D.

With the help of above theoretical properties, we now extensively search precise linear approximations with high correlations. The search process contains 4 steps.

Step 1. For each good (Ψ, Ψ') which maybe induce high correlation of σ modular addition, e.g., those listed in Appendix D, we exhaustively search 4×2^{16} mask pairs (Λ, Λ^*). Here we require that $\Gamma^{(3)} = \Lambda^*$ according to Proposition 4. The best choices of $\Lambda^*_{3,:}$ are 0x81ec5a80, 0x411af7c0, 0x2161a1e0, 0xe1970ca0 and 0xc17a8fa8. All of them may induce a large correlation $c(\sigma(\Theta), \Lambda^{(3)}, \Lambda^*)$.

Step 2. For each good candidate of $\Psi^{(1)}$, exhaustively compute the correlation for the 2^{16} pairs of (Λ', Λ) by Algorithm 1, sort and store the results. Since only the 12-th bytes of the masks are active, the search may be trivial.

Step 3. For each good candidate of $\Lambda^{(1)}$, Ψ and Ψ', run over all 2^{24} Γ', Γ and Λ. For each $(\Gamma', \Gamma, \Lambda)$, compute and sum the correlations of linear trails for all 2^8 possible $\Gamma^{(1)}$. The process involves computing $c(N3_t)$ by Algorithm 2 and computing $c(N1_t)$ and $c(N2_t)$ by Algorithm 1 and 5.

Step 4. Combining step 2 and step 3. For each Γ', Γ, Λ^*, Λ, Ψ and Ψ', find out those Λ' such that $|c_{FSM}| > 2^{-50}$, i.e., $|c(\Psi, \Psi', \Lambda, \Lambda^*, \Lambda', \Gamma, \Gamma')| > 2^{-50}$.

Consequently, we obtain 8, 127 and 957 linear approximations with absolute correlation $|c_{FSM}| > 2^{-48}$, $2^{-49} < |c_{FSM}| < 2^{-48}$ and $2^{-49} < |c_{FSM}| < 2^{-48}$. The best linear approximation we found gives $|c_{FSM}| = 2^{-47.567}$, see Appendix E.

In [12, 13], the authors give 2 linear trails with absolute correlation 2^{-48} and $2^{-49.063}$. These two trails induce two linear approximations with $|c_{FSM}| = 2^{-48.065}$, $2^{-47.760}$. The correlation of the first approximation is lower than the second, while the first linear trail has a higher correlation. They point out it is because of the aggregation effect influenced by Θ and $\Gamma^{(1)}$. However, we observed that the phenomenon is caused by the linear hull effect of S-boxes addition rather than the sum of Θ and $\Gamma^{(1)}$. We also observed that there is only one $\Gamma^{(1)}$ which dominates the overall linear approximation in above search process. Thereby, the sum for those $\Gamma^{(1)}$ in Step 3 is not necessary. Furthermore, we rebuild the automatic search model proposed in [13] and solve it by STP. We find no linear trails with ≤ 6 active S-boxes and the correlation of σ modular addition greater than $2^{-35.585}$.

5.2 Improve the Linear Approximations with 12 Active S-boxes

In [9], a linear approximation is proposed for SNOW-V$_{\boxplus_{32},\boxplus_8}$. The corresponding linear trails have 12 active S-boxes in total. The columns propagation $\Lambda^{(2)}_{i,:} \to \Lambda^*_{i,:}$ and $\Psi^{(2)}_{i,:} \to \Psi_{i,:} (0 \leq i < 4)$ all map 1 active byte to 4 active bytes.

Since we have more efficient search algorithm for S-boxes modular addition and efficient computing algorithm for σ modular addition, we may find linear approximations with higher correlation of full SNOW-V. Our search includes 4 steps.

Step 1. The starting point is from determining $\Psi^{(2)}$. According to the MDS property of AES's MixColumn matrix, if Principle 1 has higher priority, there are at least 12 active S-boxes in total. The optimal $\Psi^{(2)}$ would have one of the 4 following forms:

$$\underset{\text{(I)}}{\begin{pmatrix} x & 0 & 0 & 0 \\ 0 & 0 & 0 & x \\ 0 & 0 & x & 0 \\ 0 & x & 0 & 0 \end{pmatrix}}, \underset{\text{(II)}}{\begin{pmatrix} 0 & x & 0 & 0 \\ x & 0 & 0 & 0 \\ 0 & 0 & 0 & x \\ 0 & 0 & x & 0 \end{pmatrix}}, \underset{\text{(III)}}{\begin{pmatrix} 0 & 0 & x & 0 \\ 0 & x & 0 & 0 \\ x & 0 & 0 & 0 \\ 0 & 0 & 0 & x \end{pmatrix}}, \underset{\text{(IV)}}{\begin{pmatrix} 0 & 0 & 0 & x \\ 0 & 0 & x & 0 \\ 0 & x & 0 & 0 \\ x & 0 & 0 & 0 \end{pmatrix}},$$

where x denotes any nonzero 8-bit binary linear mask. Therefore, there are 4×255 possible choices of $\Psi^{(2)}$ as well as Ψ in this case.

Next, it is obvious that Ψ is byte-wise symmetric, namely $\sigma(\Psi) = \Psi$. Moreover, there is only one nonzero active column of $\Psi^{(1)}$ when $\Psi^{(2)}$ satisfies type (I),(II),(III) and (IV). Thereby, Λ' and Λ should also has one active column. Considering the linear hull effect of σ modular addition, the following empirical result is observed by many examples of our experiments.

Observation 2. Absolute correlation $|c(N3_t)|$ is relatively high when

$$\sigma(\Lambda^{(3)}) = \sigma(\Lambda^*) = \Psi = \Psi' = (M^{-1})^T \cdot \Psi^{(2)}. \tag{6}$$

For example, given masks $T = \sigma(U) = \sigma(V)$ which induces 12 active S-boxes for $N1_t$ and $N2_t$, we find that the correlation $|c(N3_t)|$ becomes much smaller or even zero, when $W \neq \sigma(U)$(or $\Lambda^{(3)} \neq \Lambda^*$).

Assuming that Equation (6) holds, as $\Gamma^{(3)} = \Lambda^{(3)} \oplus \Lambda$, then for each choice of $\Psi^{(2)}$, we could exhaustively search all 2^{32} possible values of $\Lambda^{(3)}_{i,:}$ satisfying $\Psi^{(1)}_{i,:} \neq \mathbf{0}$. The experiments reveal that there exist 60 $\Lambda^{(3)}, \Lambda^*, \Psi$ tuples such that $|c(N3_t)| > 2^{-43}$. The number of $\Psi^{(2)}$ with type (I), (II), (III), (IV) are 30, 14, 1 and 15 respectively. Moreover, all these $\Psi^{(2)}$ satisfy that $x = 0$xe. The maximal absolute correlation $|c(N3_t)|$ is $2^{-40.44}$ which is of type (IV). Notice that these $|c(N3_t)|$ usually have big gaps, most of them are much lower than 2^{-50}. The search space in this step is about 2^{42}.

Step 2. Since $\Lambda^{(1)}_{i,:} = \Psi^{(1)}_{i,:} = (0\text{xe}, 0\text{xe}, 0\text{xe}, 0\text{xe})$, by the previous analysis, we have $\Gamma^{(1)}_{i,:} = (y, 0\text{xe}, 0\text{xe}, 0\text{xe}), y \in \mathbb{F}^*_{2^8}$. For each y, there is a $\Psi^{(1)}_{i,:}$, $\Lambda'_{i,:}$, $\Lambda_{i,:}$ tuple corresponding to W, V, U in Equation (3), where $\Lambda_{i,:} = \Lambda^{(3)}_{i,:} \oplus (M^{-1})^T(y, 0, 0, 0)^T$. Now the task is searching for a Λ' maximizing $|c(N1_t)|$ for each y. The size of the mask space needed to be searched is about $2^{40} \times 60$. With the help of the Algorithm 1, this space could be efficiently searched within several minutes.

Step 3. Similarly, as $\Gamma^{(1)}_{i,:} = (y, 0\text{xe}, 0\text{xe}, 0\text{xe})$, we run Algorithm 5 to find good Γ' and Γ for each y. However, the experiments show that $c(N2_t)$ is very likely to be 0 when $\Gamma = \Gamma'$. Thereby, we just require that $W(\Gamma \oplus \Gamma') = 2$, i.e., they are just different in 2 bits. The size of search space is about 2^{48}. Our platform is a 64-bit CentOS workstation with Intel Xeon Sliver 2620 CPU using 48 parallel threads and AVX instructions.

Step 4. Combining these linear approximations together and summing up these approximations for each y, we will obtain a linear approximation for FSM in 3 consecutive clocks. However, we find that usually most values of y would induce a zero correlation, and there exists one y that dominates the remaining linear paths. Strictly speaking, all 2^{32} $\Lambda^{(3)}$ should also be summed up, but they are neglected due to Observation 1. Therefore, the total correlation could be approximated by that of the dominating linear path.

Finally, 257 linear approximations with absolute correlation greater than 2^{-88} and 18821 linear approximations with absolute correlation greater than 2^{-89} are found out. The best one is depicted in Table E5. It only depicts part of linear path, and the other masks can be derived by propagation rules. In [9], 864 linear approximations with $|c_{FSM}| > 2^{-92}$ for SNOW-V$_{\boxplus_{32},\boxplus_8}$ are proposed. The best one has absolute correlation $2^{-91.60}$.

Table 2 Attack complexity for 2 cases

Complexity	Case 1		Case 2	
	$l' = 233$	$l' = 236$	$l' = 233$	$l' = 237$
Time	$2^{240.86}$	$2^{243.88}$	$2^{240.86}$	$2^{244.89}$
Memory	$2^{239.80}$	$2^{238.31}$	$2^{240.37}$	$2^{238.38}$
Data	$2^{239.30}$	$2^{237.81}$	$2^{236.87}$	$2^{234.88}$

5.3 Complexity Analysis

We apply the traditional FCA for 2 cases. The first one is $|c_{FSM}| = 2^{-47.567}$ with $n = 1$ linear approximation. The second one is $|c_{FSM}| = 2^{-47.851}$ with $n = 8$ linear approximations. Let $l = 512$ denote the length of LFSR. Suppose we have collected N samples of key stream words, and the noise is folded one time to reduce the time complexity of decoding phase. Theoretically, $\binom{N}{2}$ parity check equations of the form $\bigoplus_{j=1}^{2} \boldsymbol{g}_{i_j}$ could be generated. It is expected that $M = \binom{N}{2} 2^{l'-l}$ of them would have $l - l'$ bits 0 at some indexes. It is necessary that $l'/M < c^4/(2\ln(2))$ to successfully decode by Shannon's Theorem, i.e., $N > \sqrt{2\ln(2)l'}c^{-2}2^{(l-l'+1)/2}$. Therefore, the time complexity of decoding is about $O(M + l'2^{l'})$ when FWHT technique is used. The memory complexity of generating M parity checks is about $O(\sqrt{l'2^{l-l'+3}\ln(2)}c^{-2})$ by 2-tree collision. As we have 257 linear approximations with absolute correlation greater than c, the data complexity is about $O(N/n)$.

In order to compare with [13], we give two trade-off points for each cases, see Table 2. When we require that all complexities are lower than those in [13], we could choose case 1 with $l' = 236$ or case 2 with $l' = 237$. When we prefer better trade-off point, we could choose $l' = 233$.

6 Conclusion

In this paper, we develop two new algorithms to evaluate the correlation of linear approximation of two main nonlinear components of SNOW-V efficiently, where linear hull effect is non-negligible. Since it is empirically expected that the correlation of linear approximation of S-boxes modular addition is decreasing, when evaluating the linear approximation byte by byte, we propose a depth-first prune-and-search algorithm to significantly speed up searching for linear masks with high correlation assisting with a byte(and bit) pattern matching technique. It thereby allows us to search larger mask space. This pruning strategy may be applied to other similar nonlinear structures. The second algorithm aims at computing the correlation of linear approximation of σ modular addition component, which allows us to evaluate linear approximations of SNOW-V itself rather than its variants. Inspired by addition order in σ modular addition, we also propose a technique to glue the carry information of two adjacent columns. The σ modular addition could be treated

as a GPLFM like operation in Algorithm 2. The basic idea of Algorithm 2 is also generic for other similar nonlinear components, where modulo 2^n addition could be divided into cells and then be glued together. Based on these algorithms, we extensively search precise linear approximation of full SNOW-V for different number of active S-boxes. Consequently, we find out 8, 135 and 1092 linear approximations with absolute correlation greater than $2^{-47.851}$, 2^{-49} and 2^{-50} respectively, which would derive a fast correlation attack with time/memory/data complexities $2^{240.86}$, $2^{240.37}$ and $2^{236.87}$. Our result is better than the previous results for full (or simplified) SNOW-V We also give some theoretical properties for the linear trails with 3 active S-boxes, which illustrate more information about the number, the pattern and the correlation of these linear trails, and also explains the results of the previous automatic search method. We expect our work will give a more comprehensive description of linear approximation properties for SNOW-V.

Declarations

- Funding. No funds, grants, or other support was received.
- Competing interests. The authors have no competing interests to declare that are relevant to the content of this article.

Appendix A Search for Binary Linear Approximations of $N2_t$

The technique of searching for linear approximations of $N2_t$ is similar to that of $N1_t$. Algorithm 1's pruning strategy is also applicable. One main difference is the correlation of linear approximation of duplicate S-boxes are taken into account at the same time. Let $c(U_k, E_k)$ denote the correlation of linear approximation for the S-box with input mask U_k and output mask E_k, Algorithm 5 depicts these differences more specifically.

Appendix B A Scaled Experiment to Check the Independency

Let the extension finite field $\mathbb{F}_{2^4} \cong \mathbb{F}_2[x]/(x^4 + x + 1)$, variables $X, Y, Z \in \mathbb{F}_{2^4}^3$, S^3 denotes 3 parallel S-boxes of block cipher PRESENT. Suppose that the two linear approximations are as follows.

$$\begin{aligned} &\Gamma^{(1)} \cdot (S^3(x) \boxplus_{16} y) \oplus \Gamma^{(2)} \cdot x \oplus \Gamma^{(3)} \cdot y, \\ &\Lambda^{(1)} \cdot (x \boxplus_{16} z) \oplus \Lambda^{(2)} \cdot x \oplus \Lambda^{(3)} \cdot z, \end{aligned} \tag{B1}$$

Algorithm 5 Searching for linear masks of Equation (4) with high correlation

Require: Potential binary mask Space $\mathcal{U} \times \mathcal{V} \times \mathcal{W} \times \mathcal{E}$.
Ensure: The maximal correlation c_{max} and linear masks U, V, W, E.

```
Precompute 2^24 2×2 GPLFM connection matrices, store them in memory;
c_max ← c_ini, (U_max, V_max, W_max, E_Max) ← (0,0,0,0), (c_{-1,0}, c_{-1,1}) ← (1,0);
for all possible tuples U, W, E ∈ U × W × E s.t. U^{bsf} = W^{bsf} and U, E have the same byte pattern do
    Set depth i ← 0, j ← End(U), (c_{-1,0}, c_{-1,1}) ← (1,0);
    for all possible (U_i, V_i) s.t. i ≤ j and V ∈ V do
        if i = j and V^{sf} ≠ U^{sf} then
            continue;
        end if
        Compute (c_{i,0}, c_{i,1})^t ← C_{U_i,V_i,W_i} (c_{i-1,0}, c_{i-1,1})^t;
        if i = j and |(c_{j,0} + c_{j,1}) ∏_{k=0}^{j} c(U_k, E_k)| ≥ c_max then
            c_max ← |(c_{j,0} + c_{j,1}) ∏_{k=0}^{j} c(U_k, E_k)|,
            (U_max, V_max, W_max, E_Max) ← (U, V, W, E);
        end if
        if i < j then
            if c_{i,0} = c_{i,1} = 0 or (c_{i,0} + c_{i,1} ≠ 0 and |(c_{i,0} + c_{i,1}) ∏_{j=0}^{i} c(U_j, E_j)| < c_max) then
                continue;
            else
                i ← i + 1;
            end if
        end if
    end for
end for
Return c_max, U_max, V_max, W_max, E_max.
```

where $\Lambda^{(1)} \xrightarrow{M} \Gamma^{(1)}$. M is a MDS matrix with the following form

$$\begin{pmatrix} 1 & 1 & 2 \\ 1 & 2 & 1 \\ 2 & 1 & 1 \end{pmatrix}.$$

To simplify the experiment, we require $\Lambda^{(1)} = \Lambda^{(2)} = \Lambda^{(3)}$ and $\Gamma^{(1)} = \Gamma^{(3)}$. Moreover, $\Lambda^{(1)}$ is sparse. After computing their individual probability distribution and union probability distribution, we evaluate the relative entropy between the union probability distribution and the multiplication of two individual probability distributions. The simulation results illustrate that their differences are very tiny. For example, when $\Gamma^{(1)} = \Gamma^{(3)} = (1, 0, 0, 0)$, the relative entropy is 0.007913. The reason may be that the word weight of $\Lambda^{(1)}$ and $\Gamma^{(1)}$ is not equal.

Table C1 A group of test values for Algorithm 2

Masks	Values	$\|cor\|$
U	0x050f0c08, 0x08050f0c, 0x0c08050f, 0x0f0c0805	$2^{-40.44}$
V	0x050f0c08, 0x08050f0c, 0x0c08050f, 0x0f0c0805	
W	0x050f0c08, 0x08050f0c, 0x0c08050f, 0x0f0c0805	
T	0x050f0c08, 0x08050f0c, 0x0c08050f, 0x0f0c0805	

Table C2 Empirical and theoretical absolute correlation for an example

Theoretical	Sample size	Empirical				
$2^{-13.68}$	2^{28}	$2^{-12.94}$	$2^{-14.39}$	$2^{-13.88}$	$2^{-14.35}$	$2^{-14.02}$
	2^{29}	$2^{-13.92}$	$2^{-14.02}$	$2^{-13.77}$	$2^{-13.40}$	$2^{-13.50}$
	2^{30}	$2^{-13.37}$	$2^{-14.43}$	$2^{-13.68}$	$2^{-13.77}$	$2^{-14.14}$

Appendix C Experimental Data of Algorithm 2

Table C1 indicates a group of test value for Algorithm 2. Furthermore, to verify Algorithm 2, some random sample experiments are performed. For example, let Mask $U = (0x800081c1,0x80000081,0x00000000,0x80008080)$ and $U = V = \sigma(W) = \sigma(T)$ holds. Table C2 indicates the empirical correlations by random sample experiments for this example. For each sample size, 5 empirical absolute correlations are given.

Appendix D Some good masks of σ modular addition with 3 active S-boxes

Several groups of masks with high correlation are listed in Table D3. According to subsection 5.1, we choose $\Lambda^{(3)} = \Lambda^* \oplus \Lambda$.

Appendix E Linear approximations with high correlation

We list all the 8 linear approximations with absolute correlation $> 2^{-48}$ in Table E4. We also list a linear approximation with absolute correlation $2^{-87.24}$ in Table E5.

References

[1] Clark, A., Dawson, E., Fuller, J., Golić, J., Lee, H.-J., Millan, W., Moon, S.-J., Simpson, L.: The lili-ii keystream generator. In: Batten, L., Seberry,

Table D3 Some good masks for σ modular addition

$\Lambda^{(2)}_{3,:}$	$\Lambda^{(3)}_{3,:}$	$\Lambda^{*}_{3,:}$	Λ_{12}	Θ_{12}	$\lvert c(\sigma(\Theta), \Lambda^{(3)}, \Lambda^{*})\rvert$
0x000000b7	0x81ec5a00	0x81ec5a80	0x80	0xc1,0xa1,0x91	$2^{-18}, 2^{-19}, 2^{-20}$
0x000000b7	0x81ec5ac0	0x81ec5a80	0x40	0xc1,0xa1	$2^{-19}, 2^{-20}$
0x000000b7	0x81ec5aa0	0x81ec5a80	0x20	0xc1	2^{-20}
0x0000006c	0x411af780	0x411af7c0	0x40	0x61,0x51	$2^{-19}, 2^{-20}$
0x0000006c	0x411af7e0	0x411af7c0	0x20	0x61	2^{-20}
0x00000001	0x2161a1c0	0x2161a1e0	0x20	0x31	2^{-20}
0x000000da	0xe1970ca0	0xe1970c80	0x20	0xa1	2^{-21}
0x009c0000	0xc17a8fa0	0xc17a8fa8	0x08	0xc1	2^{-22}
$\Psi^{(2)}_{3,:}$	$\Psi'_{3,:}$	$\Psi_{3,:}$		Θ_{12}	$\lvert c(\Psi, \Psi', \Theta)\rvert$
0x000000b7	0x81ec5a00	0x81ec5a80		0xc1,0xa1,0x91	$2^{-18}, 2^{-19}, 2^{-20}$
0x000000b7	0x81ec5ac0	0x81ec5a80		0xc1,0xa1	$2^{-19}, 2^{-20}$
0x000000b7	0x81ec5aa0	0x81ec5a80		0xc1	2^{-20}
0x000000b7	0xc1ec5a00	0x81ec5a80		0x81,0xe1,0xd1	$2^{-18}, 2^{-19}, 2^{-20}$
0x000000b7	0xc1ec5ac0	0x81ec5a80		0x81,0xe1	$2^{-19}, 2^{-20}$
0x000000b7	0xc1ec5aa0	0x81ec5a80		0x81	2^{-20}
0x000000b7	0xa1ec5a00	0x81ec5a80		0x81,0xb1	$2^{-19}, 2^{-20}$
0x000000b7	0xa1ec5ac0	0x81ec5a80		0x81	2^{-20}
0x000000b7	0xe1ec5a00	0x81ec5a80		0xc1,0xf1	$2^{-19}, 2^{-20}$
0x000000b7	0xe1ec5ac0	0x81ec5a80		0xc1	2^{-20}
0x000000b7	0x91ec5a00	0x81ec5a80		0x81	2^{-20}
0x000000b7	0xb1ec5a00	0x81ec5a80		0xa1	2^{-20}
0x000000b7	0xd1ec5a00	0x81ec5a80		0xc1	2^{-20}
0x000000b7	0xf1ec5a00	0x81ec5a80		0xe1	2^{-20}
0x0000006c	0x411af780	0x411af7c0		0x61,0x51	$2^{-19}, 2^{-20}$
0x0000006c	0x411af7e0	0x411af7c0		0x41	2^{-20}
0x0000006c	0x611af780	0x411af7c0		0x41,0x71	$2^{-19}, 2^{-20}$
0x0000006c	0x611af7e0	0x411af7c0		0x41	2^{-20}
0x0000006c	0x511af780	0x411af7c0		0x41	2^{-20}
0x0000006c	0x711af780	0x411af7c0		0x61	2^{-20}
0x00000001	0x2161a1c0	0x2161a1e0		0x31	2^{-20}
0x00000001	0x3161a1c0	0x2161a1e0		0x21	2^{-20}

J. (eds.) Information Security and Privacy, pp. 25–39. Springer, Berlin, Heidelberg (2002)

[2] Jönsson, F., Johansson, T.: A fast correlation attack on lili-128. Information Processing Letters **81**(3), 127–132 (2002)

[3] Courtois, N.T., Meier, W.: Algebraic attacks on stream ciphers with linear feedback. In: Biham, E. (ed.) Advances in Cryptology — EUROCRYPT 2003, pp. 345–359. Springer, Berlin, Heidelberg (2003)

Table E4 Some good linear approximations of full SNOW-V

Γ'_{12}	Γ_{12}	Λ'_{12}	Λ_{12}	$\Lambda^*_{3,:}$	$\Psi_{3,:}$	$\Psi'_{3,:}$	$\lvert c_{FSM}\rvert$
0x08	0x0c	0x80	0x80	0x81ec5a80	0x81ec5a80	0x81ec5a00	$2^{-47.579}$
0x08	0x0c	0x40	0x40	0x81ec5a80	0x81ec5a80	0x81ec5a00	$2^{-47.567}$
0x0d	0x0d	0x80	0x80	0x81ec5a80	0x81ec5a80	0x81ec5a00	$2^{-47.772}$
0x0d	0x0d	0x40	0x40	0x81ec5a80	0x81ec5a80	0x81ec5a00	$2^{-47.760}$
0x30	0x20	0x80	0x80	0x81ec5a80	0x81ec5a80	0x81ec5a00	$2^{-47.672}$
0x30	0x20	0x40	0x40	0x81ec5a80	0x81ec5a80	0x81ec5a00	$2^{-47.660}$
0x78	0x78	0x80	0x80	0x81ec5a80	0x81ec5a80	0x81ec5a00	$2^{-47.851}$
0x78	0x78	0x40	0x40	0x81ec5a80	0x81ec5a80	0x81ec5a00	$2^{-47.839}$

Table E5 A binary linear approximation with $\lvert c_{FSM}\rvert = 2^{-87.24}$ of 12 active S-boxes

$\Lambda_{0,:}$	$\Lambda'_{0,:}$	$\Gamma^{(2)}_{0,:}$	$\Gamma_{0,:}$	$\Gamma'_{0,:}$	$\Lambda^{(2)}_{0,:}$
0x1114190d	0x191c1189	0x0e0e0e1c	0x20094187	0x300d4187	0x0e0e0e0e

[4] Ekdahl, P., Johansson, T.: Snow-a new stream cipher. In: Proceedings of First Open NESSIE Workshop, KU-Leuven, pp. 167–168 (2000)

[5] Ekdahl, P., Johansson, T.: A new version of the stream cipher snow. In: Nyberg, K., Heys, H. (eds.) Selected Areas in Cryptography, pp. 47–61. Springer, Berlin, Heidelberg (2003)

[6] UEA2&UIA, I.: Specification of the 3GPP Confidentiality and Integrity Algorithms UEA2& UIA2. Document 2: SNOW 3G Specifications. Version: 1.1. ETSI. SAGE Specification (2006)

[7] Ekdahl, P., Johansson, T., Maximov, A., Yang, J.: A new SNOW stream cipher called SNOW-V. IACR Trans. Symmetric Cryptol. **2019**(3), 1–42 (2019)

[8] Nyberg, K., Wallén, J.: Improved linear distinguishers for snow 2.0. In: Robshaw, M. (ed.) Fast Software Encryption, pp. 144–162. Springer, Berlin, Heidelberg (2006)

[9] Gong, X., Zhang, B.: Resistance of SNOW-V against fast correlation attacks. IACR Trans. Symmetric Cryptol. **2021**(1), 378–410 (2021)

[10] Yang, J., Johansson, T., Maximov, A.: New improved attacks on SNOW-V. Cryptology ePrint Archive, Report 2021/544. https://eprint.iacr.org/2021/544 (2021)

[11] Hoki, J., Isobe, T., Ito, R., Liu, F., Sakamoto, K.: Distinguishing and Key

Recovery Attacks on the Reduced-Round SNOW-V and SNOW-Vi. Cryptology ePrint Archive, Report 2021/546. https://eprint.iacr.org/2021/546 (2021)

[12] Shi, Z., Jin, C., Zhang, J., Cui, T., Ding, L.: A Correlation Attack on Full SNOW-V and SNOW-Vi. Cryptology ePrint Archive, Report 2021/1047. https://ia.cr/2021/1047 (2021)

[13] Shi, Z., Jin, C., Jin, Y.: Improved Linear Approximations of SNOW-V and SNOW-Vi. Cryptology ePrint Archive, Report 2021/1105. https://ia.cr/2021/1105 (2021)

[14] Yang, J., Johansson, T., Maximov, A.: Vectorized linear approximations for attacks on SNOW 3g. IACR Trans. Symmetric Cryptol. **2019**(4), 249–271 (2019)

[15] Zhang, B., Xu, C., Meier, W.: Fast correlation attacks over extension fields, large-unit linear approximation and cryptanalysis of snow 2.0. In: Gennaro, R., Robshaw, M. (eds.) Advances in Cryptology – CRYPTO 2015, pp. 643–662. Springer, Berlin, Heidelberg (2015)

[16] Meier, W., Staffelbach, O.: Fast correlation attacks on certain stream ciphers. J. Cryptol. **1**(3), 159–176 (1989)

[17] Johansson, T., Jönsson, F.: Fast correlation attacks based on turbo code techniques. In: Wiener, M.J. (ed.) Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings. Lecture Notes in Computer Science, vol. 1666, pp. 181–197. Springer, Berlin, Heidelberg (1999)

[18] Johansson, T., Jönsson, F.: Fast correlation attacks through reconstruction of linear polynomials. In: Bellare, M. (ed.) Advances in Cryptology — CRYPTO 2000, pp. 300–315. Springer, Berlin, Heidelberg (2000)

[19] Canteaut, A., Trabbia, M.: Improved fast correlation attacks using parity-check equations of weight 4 and 5. In: Preneel, B. (ed.) Advances in Cryptology — EUROCRYPT 2000, pp. 573–588. Springer, Berlin, Heidelberg (2000)

[20] Clark, A., Golić, J.D., Dawson, E.: A comparison of fast correlation attacks. In: Gollmann, D. (ed.) Fast Software Encryption, pp. 145–157. Springer, Berlin, Heidelberg (1996)

[21] Mihaljevi, M.J., Fossorier, M.P.C., Imai, H.: Fast correlation attack algorithm with list decoding and an application. In: Matsui, M. (ed.) Fast Software Encryption, pp. 196–210. Springer, Berlin, Heidelberg (2002)

[22] Mihaljević, M.J., Golić, J.D.: Convergence of a bayesian iterative error-correction procedure on a noisy shift register sequence. In: Rueppel, R.A. (ed.) Advances in Cryptology — EUROCRYPT' 92, pp. 124–137. Springer, Berlin, Heidelberg (1993)

[23] Chose, P., Joux, A., Mitton, M.: Fast correlation attacks: An algorithmic point of view. In: Knudsen, L.R. (ed.) Advances in Cryptology — EUROCRYPT 2002, pp. 209–221. Springer, Berlin, Heidelberg (2002)

[24] Todo, Y., Isobe, T., Meier, W., Aoki, K., Zhang, B.: Fast correlation attack revisited. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018, pp. 129–159. Springer, Cham (2018)

[25] Yang, J., Johansson, T., Maximov, A.: Spectral analysis of ZUC-256. IACR Trans. Symmetric Cryptol. **2020**(1), 266–288 (2020)

[26] Maximov, A., Johansson, T.: Fast computation of large distributions and its cryptographic applications. In: Roy, B. (ed.) Advances in Cryptology - ASIACRYPT 2005, pp. 313–332. Springer, Berlin, Heidelberg (2005)