An Improved Estimate of the Correlation of Distinguisher for Dragon

Joo Yeon Cho

Helsinki University of Technology,
Laboratory for Theoretical Computer Science,
P.O. Box 5400, FI-02015 TKK, Finland
joo.cho@tkk.fi

Abstract. The function F of Dragon plays an important role on both the keystream generation and the internal state update. We analyze the function F of Dragon by linear cryptanalytic methods. Thanks to an efficient algorithm on linear approximations of the modular addition, we observed that there were a large number of approximations with significant correlations in the nonlinear components of the function F. Using linear approximations of the modular addition with correlations up to 2^{-20}, we estimate that the distinguisher for Dragon has a correlation of $2^{-66.45}$.
Keywords : Stream Ciphers, eSTREAM, Dragon, Modular Addition, Distinguishing Attacks.

1 Introduction

In 2004, the ECRYPT project launched a new multi-year project eSTREAM, the ECRYPT Stream Cipher project, to identify new stream ciphers that might become suitable for widespread adoption as an international industry standard [**?**]. In Phase 3 of eSTREAM, eight stream ciphers have been selected as focus stream ciphers in the software category (Profile 1) and Dragon is one of those stream ciphers.

Dragon is a word oriented stream cipher. It has a 1024-bit internal state and a 64-bit internal memory.

It employs a function F which is a carefully designed nonlinear function whose output is used for the internal state update and the keystream generation. Hence, the cryptographic strength of the function F is one of important assumptions for the Dragon stream cipher.

In this paper, the function F of Dragon is analyzed by linear cryptanalytic methods. First, the nonlinear components of the function F such as modular addition and S-boxes are analyzed and their linear approximations are derived. Second, we observed that there were a large number of approximations with significant correlations in the components. To achieve an accurate estimate for the correlation of the function F, one should take multiple linear chains among the components into account.

The first distinguisher for the Dragon stream cipher was reported in [**?**] and then, its efficiency was improved in [**?**]. Here in this paper, we will show that the correlation of the distinguisher, which was reported in [**?**] as $2^{-75.32}$, was quite underestimated. Using the efficient algorithm for linear approximations of the modular addition that was developed by Wallen [**?**], we were able to search the linear approximations of the components extensively. We investigated correlations of the components by using linear approximations of the modular addition with correlations up to 2^{-20}. As a result, our computer simulation shows that the correlation of our distinguisher is around $2^{-66.45}$. Therefore, we claim that the Dragon

stream cipher is distinguishable from a random cipher by observing around 2^{133} keystream words under the assumption that 2^{59} memory bits are guessed. This is the best cryptanalytic result for Dragon in the open literature.

This paper is organized as follows. Section 2 presents a brief description of the function F of Dragon. In Section 3, the correlations of nonlinear components of Dragon are discussed. And then, a set of linear approximations of the function F are derived and a distinguisher is built by combining these approximations in Section 4. Section 5 concludes the work.

2 The Function F of Dragon

The function F is the nonlinear state update function of Dragon [**?**]. The function F defined in Dragon is a nonlinear function that plays double role, first it updates the internal state of the cipher and its second role is to generate the keystream. The function F takes six 32-bit words, denoted as (a, b, c, d, e, f), as the input and produces six 32-bit words, denoted as (a', b', c', d', e', f'), as the output. Among the six output words, two words (b', c') are used as new state words, two words (a', e') are the output of a 64-bit keystream word and the remaining two words (d', f') are discarded.

The function F consists of xors, modular additions, and six nonlinear functions that are called as G_1, G_2, G_3, H_1, H_2 and H_3. The detailed structure of the function F is shown in Figure 1. Note that all the components are operated in $GF(2^{32})$. The essential components

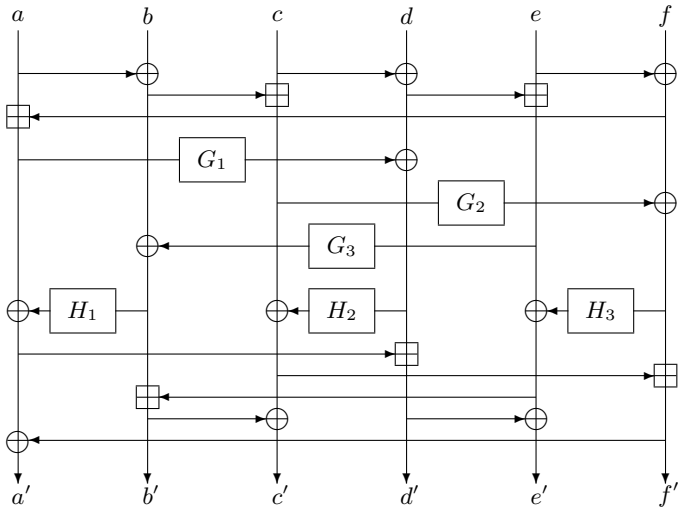

Fig. 1. F function

of the functions G and H are the two S-boxes: S_1 and S_2. Both S_1 and S_2 transform 8-bit inputs into 32-bit outputs, that is, $\mathbb{Z}_{2^8} \rightarrow \mathbb{Z}_{2^{32}}$. The functions G and H use both S-boxes selectively four times. Hence, G and H take a 32-bit input and produce a 32-bit output, that is, $G, H : \mathbb{Z}_{2^{32}} \rightarrow \mathbb{Z}_{2^{32}}$. The structure of the functions G and H is described as follows:

$$G_1(x) = S_1(x_0) \oplus S_1(x_1) \oplus S_1(x_2) \oplus S_2(x_3)$$

$$G_2(x) = S_1(x_0) \oplus S_1(x_1) \oplus S_2(x_2) \oplus S_1(x_3)$$
$$G_3(x) = S_1(x_0) \oplus S_2(x_1) \oplus S_1(x_2) \oplus S_1(x_3)$$
$$H_1(x) = S_2(x_0) \oplus S_2(x_1) \oplus S_2(x_2) \oplus S_1(x_3)$$
$$H_2(x) = S_2(x_0) \oplus S_2(x_1) \oplus S_1(x_2) \oplus S_2(x_3)$$
$$H_3(x) = S_2(x_0) \oplus S_1(x_1) \oplus S_2(x_2) \oplus S_2(x_3)$$

where the 32-bit input x is divided into four bytes x_i; $i = 0, 1, 2, 3$; such that $x = (x_0, x_1, x_2, x_3)$. The byte x_0 denotes the most significant byte of the word.

Using the F function, the keystream is generated as follows. Note that the states of a nonlinear shift register is denoted as $B_0, B_1, \ldots, B_{31}$ where B_i is a 32-bit word and an internal memory is denoted by $M = (M_L||M_R)$, where M_L and M_R are 32-bit words, respectively.

1. Input : $\{B_0, B_1, \ldots, B_{31}\}$ and $M = (M_L||M_R)$
2. $a = B_0, b = B_9, c = B_{16}, d = B_{19}, e = B_{30} \oplus M_L, f = B_{31} \oplus M_R$.
3. $(a', b', c', d', e', f') = F(a, b, c, d, e, f)$
4. $B_0 = b', B_1 = c'$ and $B_i = B_{i-2}, 2 \leq i \leq 31, M = M + 1$
5. Output : $k = (a'||e')$

For a complete description of Dragon, we refer the reader to the paper [**?**].

3 Components of Function *F*

Let n be a non-negative integer. Given two vectors $x = (x_0, \ldots, x_{n-1})$ and $y = (y_0, \ldots, y_{n-1})$ where $x, y \in GF(2^n)$, let $x \cdot y$ denote a standard inner product defined as $x \cdot y = x_0 y_0 \oplus \ldots \oplus x_{n-1} y_{n-1}$. A linear mask is a constant vector that is used to compute an inner product of a n-bit string.

Let us assume that we have a function $f : \{0,1\}^m \to \{0,1\}^n$ for some positive integers m and n. Given a linear input mask $\Lambda \in GF(2^m)$ and a linear output mask $\Gamma \in GF(2^n)$, the correlation of an approximation $\Lambda \cdot x = \Gamma \cdot f(x)$ is measured as follows.

$$\epsilon_f(\Lambda, \Gamma) = 2^{-n}(\#(\Lambda \cdot x \oplus \Gamma \cdot f(x) = 0) - \#(\Lambda \cdot x \oplus \Gamma \cdot f(x) = 1))$$

where $x \in GF(2^m)$ and runs through all possible values. Then, $Pr[\Lambda \cdot x = \Gamma \cdot f(x)] = \frac{1}{2}(1 + \epsilon_f(\Lambda, \Gamma))$. Specially, the correlation on modular addition $\boxplus : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is denoted as $\epsilon_+(\Lambda_1, \Lambda_2, \Gamma)$ where both Λ_1, Λ_2 are input masks and $\Gamma \in GF(2^n)$ is an output mask.

3.1 Approximations of Modular Addition and S-boxes

We discuss the modular addition and the S-boxes that are the elementary nonlinear components of the function F. Then, linear approximations on the function F are developed.

Modular addition We consider a modular addition with two inputs operated in $GF(2^{32})$. In the paper of [**?**], Wallen developed an efficient algorithm to generate a whole set of linear approximations of the modular addition where their correlations are given. Table 1 contains results obtained using Wallen's algorithm and shows the relation between the correlations

of modular addition and the number of all linear approximations that hold such correlation. This algorithm is specially useful to find linear approximations when modular additions are combined with other nonlinear components such as S-boxes. We will discuss this in the next section. In [**?**], Nyberg and Wallen presented a generalized algorithm that produces linear approximations when the modular addition has arbitrary number of inputs. It is important to note that linear approximations with low Hamming weights tend to be highly correlated.

correlation	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
# of linear approx.	2^{8}	$2^{13.9}$	$2^{19.1}$	$2^{23.9}$	$2^{28.4}$	$2^{30.7}$	$2^{28.0}$

Table 1. Correlations and the number of linear approximations of modular addition

The S-boxes S_1 and S_2 Suppose we try to find linear approximations of S_1 and S_2 by exhaustive search. Since S_1 and S_2 are 8×32 S-boxes, the total number of linear approximations for each S-box becomes $2^{8+32} - 1 = 2^{40} - 1$. The correlation of each linear approximation is also determined by feeding 2^8 inputs on the S-boxes.

However, due to the structure of the function F, we are interested in some linear approximations that hold with strong biases over both S-boxes (and thereby, the functions G and H) and the modular addition. This means that "good" linear approximations of the S-boxes, if they exist, could be found among the linear approximations of modular addition and choose one by one starting from the approximations with the highest biases Table 2 shows an example of such linear approximations.

S-box	ϵ_S	ϵ_+
S_1	$\epsilon_{S_1}(0, \text{0x61300000}) = -2^{-1.83}$	$\epsilon_+(\text{0x41200000}, \text{0x41a00000}, \text{0x61300000}) = 2^{-3}$
S_2	$\epsilon_{S_2}(0, \text{0x60020300}) = -2^{-1.83}$	$\epsilon_+(\text{0x40020200}, \text{0x40030200}, \text{0x60020300}) = 2^{-3}$

Table 2. An example of linear approximations of S_1 and S_2

3.2 Functions G, H and Modular Addition

According to the structure of the function F, there are two types of combinations between modular addition and the function G or H. Both types are depicted in Figure 2. In Type 1, the output z is determined by a linear addition of the output of modular addition and that of the function T. Hence, the correlation of $\Lambda_1 \cdot x \oplus \Lambda_2 \cdot y = \Gamma \cdot z$ is computed as follows.

$$\epsilon_1(\Lambda_1, \Lambda_2, \Gamma) = \epsilon_+(\Lambda_1 \cdot x \oplus \Lambda_2 \cdot y = \Gamma \cdot (x \boxplus y))\epsilon_T(\Gamma \cdot T(r) = 0)$$

For example, in the function F, a linear approximation such as $\Lambda_1 \cdot (a \oplus b) \oplus \Lambda_2 \cdot c = \Gamma \cdot (b' \oplus c')$ is a Type 1 combination.

On the other hand, in Type 2, modular addition and the function T are dependent since the output of modular addition becomes the input of the function T. Hence, by the correlation

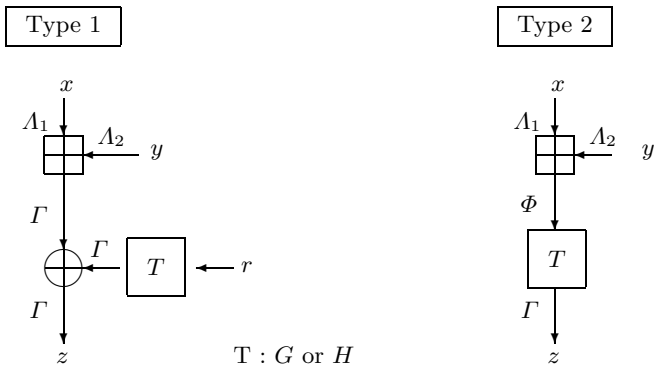

Fig. 2. Two types of combinations between modular addition and the function G or H

theory [**?**], the correlation of $\Lambda_1 \cdot x \oplus \Lambda_2 \cdot y = \Gamma \cdot z$ is equal to a sum of all the partial correlations as follows.

$$\epsilon_2(\Lambda_1, \Lambda_2, \Gamma) = \sum_{\Phi} \epsilon_+(\Lambda_1 \cdot x \oplus \Lambda_2 \cdot y = \Phi \cdot (x \boxplus y))\epsilon_T(\Phi \cdot (x \boxplus y) = \Gamma \cdot z).$$

Table 3 exemplifies how the dependency affects the correlation of linear approximations. In next section, we search the linear approximations of the function F by extending the

type	components	Γ	$\epsilon_+(\Gamma,\Gamma,\Gamma)$	$\epsilon_T(\Gamma,\Gamma)$	$\epsilon_+ \times \epsilon_T$	$\epsilon_i(\Gamma,\Gamma,\Gamma)$
Type 1	$\boxplus$, H	0x0x0600018d	2^{-3}	$-2^{-8.58}$	$-2^{-11.58}$	$-2^{-11.58}$
Type 2	$\boxplus$, G_1	0x0x30303001	2^{-3}	$2^{-10.66}$	$2^{-13.66}$	$2^{-12.19}$
Type 2	$\boxplus$, G_2	0x0x28018001	2^{-3}	$2^{-10.44}$	$2^{-13.44}$	$2^{-12.96}$
Type 2	$\boxplus$, G_3	0x0x60006002	2^{-3}	$-2^{-11.22}$	$-2^{-14.22}$	$-2^{-13.51}$

Table 3. Correlations of linear approximations of Type 1 and 2

dependency of the components.

4 Linear Approximation of Function F and Distinguisher

As described earlier, the function F transforms a 6-word string (a, b, c, d, e, f) into a 6-word string (a', b', c', d', e', f') by using xors, modular additions, the function G and H. According to the structure of F shown in Figure 1, six output words of the function F are expressed as a function of input words in the following way.

$$\begin{aligned}
a' &= [(a \boxplus (e \oplus f)) \oplus H_1] \oplus [(e \oplus f \oplus G_2) \boxplus (H_2 \oplus ((a \oplus b) \boxplus c))] \\
b' &= [(a \oplus b \oplus G_3] \boxplus [((c \oplus d) \boxplus e) \oplus H_3] \\
c' &= [(a \oplus b) \boxplus c] \oplus H_2 \oplus b' \\
d' &= (c \oplus d \oplus G_1) \boxplus ([a \boxplus (e \oplus f)] \oplus H_1)
\end{aligned}$$

$$e' = [((a \boxplus (e \oplus f)) \oplus H_1) \boxplus (c \oplus d \oplus G_1)] \oplus [H_3 \oplus ((c \oplus d) \boxplus e)]$$
$$f' = [(e \oplus f \oplus G_2) \boxplus [(a \oplus b) \boxplus c] \oplus H_2]$$

If we apply repeatedly the following three types of linear approximation:

$$\Upsilon_1 \cdot (x \boxplus y) = \Upsilon_2 \cdot x \oplus \Upsilon_3 \cdot y$$
$$\Upsilon_4 \cdot G(a) = \Upsilon_5 \cdot a$$
$$\Upsilon_6 \cdot H(b) = 0$$

Type 1 and 2 then, each output word of the function F can be approximated by input words in two types that are shown in Figure 3. The approximations of the output words a', c' and e' are classified into type A and the others are type B. Hence, we can compute the correlations

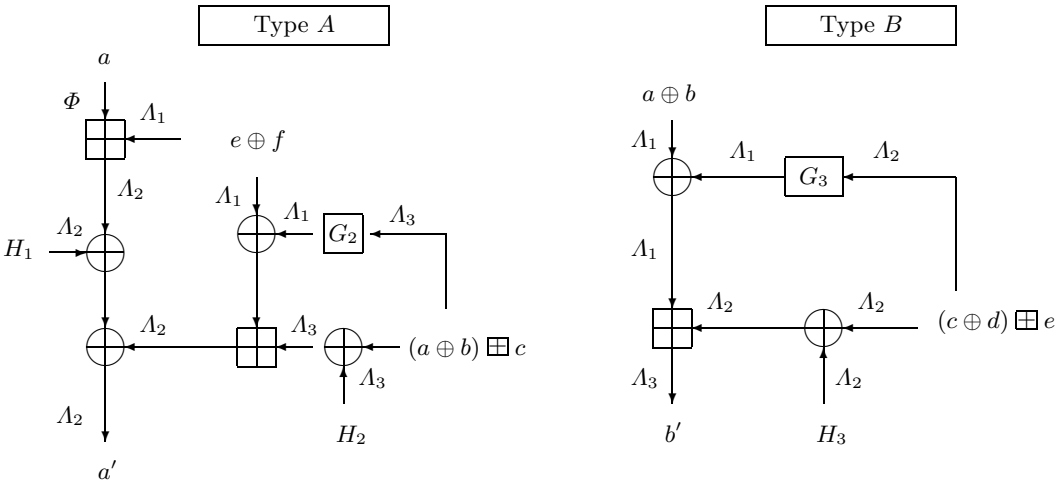

Fig. 3. Linear approximations of the function F

of two types of approximations by counting linear approximations of the components and summing up their correlations. For example, the correlation of $\Phi \cdot a = \Lambda_2 \cdot a'$ listed in the Table 4 is expressed as follows:

$$\epsilon_a(\Phi, \Lambda_2) = \epsilon_{H_1}(0, \Lambda_2) \sum_{\Lambda_1} \epsilon_+(\Phi, \Lambda_1, \Lambda_2) \sum_{\Lambda_3} \epsilon_+(\Lambda_1, \Lambda_3, \Lambda_2) \epsilon_{G_2}(\Lambda_1, \Lambda_3) \epsilon_{H_2}(0, \Lambda_3). \quad (1)$$

Due to the symmetric structure of the function F, the correlations of $\Phi \cdot e = \Lambda_2 \cdot e'$ can be computed by substituting H_1, H_2 and G_2 for H_3, H_1 and G_1 functions, respectively. By experiments, we searched the best linear approximation of each output word of the function F and the results are listed in Table 4. We note that the approximations of b' and c' are not used for the distinguisher. It is an open question, however, whether these linear approximation can be used for other attacks. Since d' and f' are not reused in next cycle, we omit their linear approximations.

type	linear approximation	correlation
A	$0x1800000d \cdot a' = 0x1800000d \cdot a$	$-2^{-38.47}$
B	$0x0600018d \cdot b' = 0x0600018d \cdot (a \oplus b)$	$2^{-21.47}$
A	$0x0600018d \cdot c' = 0x0600018d \cdot c$	$-2^{-36.04}$
A	$0x0600018d \cdot e' = 0x0600018d \cdot e$	$2^{-36.69}$

Table 4. Best linear approximations of the function F

4.1 Distinguisher

From the results discussed in the previous section, a distinguisher for Dragon can be easily derived by combining two linear approximations of the function F. According to the state update rule, the following relation holds:

$$B_0[t] = B_{30}[t+15], \quad t = 0, 1, \ldots. \tag{2}$$

Since a keystream word a and e correspond to B_0 and $B_{30} \oplus M_L$, respectively, it is easy to see that a distinguisher is constructed as

$$\Gamma \cdot a'[t] \oplus \Gamma \cdot e'[t+15] = \Gamma \cdot a[t] \oplus \Gamma \cdot e[t+15] = \Gamma \cdot M_L[t+15]. \tag{3}$$

By guessing the initial value of $M[0] = M_L[0]||M_R[0]$, we can calcuate $M_L[t+15]$ according to the memory update rule. Note that we do not need to guess the whole 64-bit $M[0]$ since Γ = 0x0600018d does not use five most significant bits. Hence, we need to guess 27 bits of $M_L[0]$ and 32 bits of $M_R[0]$. For correctly guessed $M[0]$ (and thereby, $M_L[t+15]$), $\Gamma \cdot a'[t] \oplus \Gamma \cdot e'[t+15]$ shows an estimated correlation. This enables us to distinguish Dragon from a random function. Moreover, this means that a 59-bit initial value of the memory M can be retrieved by using our distinguisher.

We note that this distinguisher was also presented in the paper [**?**] without the improvement of the correlation. However, thanks to the efficiency of our implementation, the linear approximations of the nonlinear components of function F have been extensively searched. The collection of these approximation is of Type A and shown in Figure 3.

First, we generate a pair of input masks (Φ, Λ_1) and an output mask Λ_2 of modular addition holding with the correlation of up to 2^{-20}. Then, given Φ and Λ_2, we generate again all input masks Λ_1 of modular addition satisfying $\epsilon_+(\Phi, \Lambda_1, \Lambda_2) \geq 2^{-20}$. Then, given Λ_1 and Λ_2, we generate all input masks Λ_3 of modular addition satisfying $\epsilon_+(\Lambda_1, \Lambda_3, \Lambda_3) \geq 2^{-20}$. For each Λ_1 and Λ_3, we compute $\epsilon_{H_2}(0, \Lambda_3)$ and $\epsilon_{G_2}(\Lambda_3, \Lambda_1)$, respectively. Hence, we can compute $\sum_{\Lambda_3} \epsilon_+(\Lambda_1, \Lambda_3, \Lambda_2)\epsilon_{H_2}(0, \Lambda_3)\epsilon_{G_2}(\Lambda_3, \Lambda_1)$. We can also compute $\epsilon_{H_1}(0, \Lambda_2)$ using Λ_2. Hence, for given Φ and Λ_2, we compute the correlation of $\Phi \cdot a = \Gamma_2 \cdot a'$ according to Equation (1). We repeat this procedure to compute the correlation of $\Phi \cdot e = \Gamma \cdot e'$ by choosing new linear masks and replacing H_1, H_2 and G_2 by H_3, H_1 and G_1, respectively. Thus, we get the correlation

$$\epsilon_e(\Phi, \Lambda_2) = \epsilon_{H_3}(0, \Lambda_5) \sum_{\Lambda_4} \epsilon_+(\Phi, \Lambda_4, \Lambda_5) \sum_{\Lambda_6} \epsilon_+(\Lambda_4, \Lambda_6, \Lambda_5)\epsilon_{H_1}(0, \Lambda_6)\epsilon_{G_1}(\Lambda_6, \Lambda_4).$$

Finally, we can compute more accurate correlation of Distinguisher (3) which is denoted by ϵ_D, by the following equation:

$$\epsilon_D = \sum_{\Phi} \epsilon_a(\Phi, \Lambda_2)\epsilon_e(\Phi, \Lambda_2). \tag{4}$$

We searched the mask Λ_2 that could build the best correlation of the distinguisher. The result of computer simulation is displayed in Table 5. We can see that $\epsilon_D > \epsilon_a(\Lambda_2, \Lambda_2) \times \epsilon_e(\Lambda_2, \Lambda_2)$ due to a large number of approximations with diffrent Φ.

Λ_2	$\epsilon_a(\Lambda_2, \Lambda_2)$	$\epsilon_e(\Lambda_2, \Lambda_2)$	ϵ_D
0x0600018d	$2^{-39.35}$	$2^{-36.69}$	$2^{-66.45}$

Table 5. Correlation of the distinguisher

4.2 Experiments

We performed a few experiments to verify our results. First, we investigated the correlation of $a'[t]$ and $e'[t+15]$. We counted the number of linear approximations of modular addition that have a correlation with up to 2^{-20} when the output mask is set to 0x0600018d. We found that there are around $2^{24.6}$ such pairs. Since for both a'[t] and e'[t=15], we have used Type A approximations as shown in Figure 3, it is expected that many input mask pairs , which correspond to Φ and Γ_1, affect the correlation of the distinguisher in a non-negligible way.

Next, we verified the correctness of our implementation by testing a simple combination of the modular addition and the function G_1. We set two input masks of the modular addition and an output mask of the function G_1 by 0x30303001, as listed in Table 3, and examined its real correlation by experiments. Figure 4 shows that our estimation is close to the experimental result.

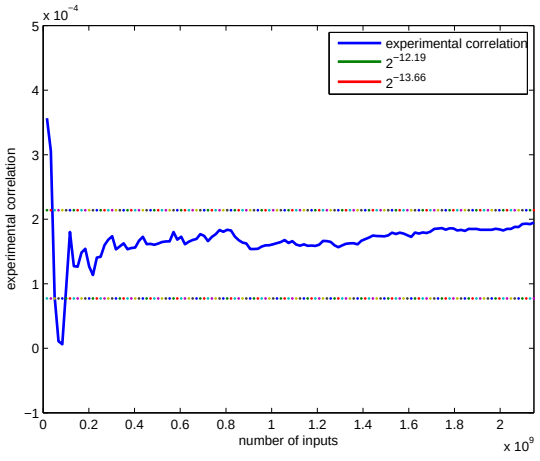

Fig. 4. Experimental results on the correlation of the components of the function F

5 Conclusion

In this paper, we analyzed the function F of Dragon by linear cryptanalytic methods. We found that the correlation of the linear distinguisher that was reported in the previous research was underestimated. We computed an accurate correlation of the distinguisher by searching extensively the linear approximations with significant correlations. Our result showed that the correlation is higher than the previous result by a factor of 2^9. Our technique can be also used as a tool to analyze other nonlinear functions that retain correlations with linear functions.

Acknowledgment

I wish to thank Kaisa Nyberg and Josef Pieprzyk for their useful comments that helped to improve the paper. I also wish to thank Risto Hakala for providing an efficient code and useful comments for experiments.