Finding Good Linear Approximations of Block Ciphers and its Application to Cryptanalysis of Reduced Round DES

Rafaël Fourquet[1], Pierre Loidreau[2] and Cédric Tavernier[3]

[1] University of Paris 8 Saint-Denis, France
[2] CELAR, France
[3] Communication et Systèmes, France

Abstract. In this paper we design an algorithm determining the list of linear approximations of a m-variate Boolean function within a given bias. We show how to adapt this algorithm in order to find multiple approximations of 8 rounds of the DES with biases of the same order as the best bias obtained by Matsui. We propose a new very efficient resulting attack based on a soft decision decoding technique of first order Reed-Muller codes.

Keywords : Linear cryptanalysis, Reed-Muller codes, coding theory, DES, multiple linear approximations.

1 Introduction

Since it was designed by Matsui in 1993 [20] and its success in the cryptanalysis of the DES [21], linear cryptanalysis has become a powerful tool in the analysis of block ciphers. Now conceivers of block ciphers have at least to prove that their cipher is immune to linear cryptanalysis.

One of the crucial steps of linear cryptanalysis in terms of time and memory complexity consists of the quantity of plaintext-ciphertext pairs (afterward denoted data-complexity) required so that the attack succeeds with a good probability. This data-complexity can be derived from linear relations involving key bits, plaintext and ciphertext bits. Suppose that the attacker obtained such a relation which is satisfied with a bias $1/2+\varepsilon$ or $1/2-\varepsilon$, then this data complexity is proportional to $1/\varepsilon^2$. Namely, Matsui proved that a data-complexity of

$$N = 1/\varepsilon^2 \approx 2^{43}$$

was sufficient to recover the key of a 16-rounds DES with a probability of 85% ([20]) using 2^{43} evaluations of the DES. To obtain this result Matsui derived the best linear relation between key bits, plaintext and ciphertext bits on 14 round of the DES which is satisfied with a bias $\varepsilon = -1.19 \times 2^{-21}$. More recently, Junod showed that with an available data-complexity of 2^{43} the complexity of

the attack had been overestimated by M. Matsui, and that 2^{41} evaluations of the DES were enough to succeed in 85% of the cases, [14].

In 1994, Kaliski and Robshaw showed that the knowledge of several linear relations involving the same key bits and with biases of the same order could reduce significantly the data-complexity of the attack ([18]). They experimentally applied their analysis to five rounds of the DES, used two linear relations involving the same key bits and with biases of the same order. They showed that the data-complexity was in that case divided by two compared to the case where a single relation is used.

The constraint on the key bits has been erased by Biryukov, De Cannière and Quisquater who showed how to use multiple linear relations to diminish the data-complexity, [1]. They showed that if there are n statistically independent linear relations involving key, plaintext and ciphertext bits satisfied with respective biases ε_j, for $j = 1, \ldots, n$ then the data-complexity N becomes

$$N \approx 1/\sum_{j=1}^{n} \varepsilon_j^2$$

Murphy showed that a part of the analysis was wrong, and that this theoretical approximation was no longer valid in the case where, in the relations, there are linear dependencies between plaintext and ciphertext bits, [22]. However, experimentation on reduced-round versions of the AES-candidate Serpent showed that the statistical independence is fulfilled whenever the number n of available approximation is not too important, [5].

Very recently, a model coming from the field of information theory was applied to the problem, [4, 7]. In this work, the authors suppose that an attacker has obtained n linear relations between key, plaintext and ciphertext bits satisfied with respective biases ε_j. Suppose moreover that the vector space spanned by the key bits has dimension k. Then the problem of finding k key bits is modeled into a problem of decoding a random code of dimension k and length n over a Gaussian channel where the noise depends on the value of ε_j for $j = 1, \ldots, n$. Different decoding algorithms can be employed and results were obtained showing how to recover 22 bits of the key with a probability of success 50% with a data-complexity of 2^{20}.

All these results point out how crucial it is to be able to compute multiple linear relations between key, plaintext and ciphertext bits which are satisfied with the best possible biases. Originally Matsui obtained his equations by exploiting a bias in the S-boxes and chaining the probabilities. The so-called *piling-up lemma* gives the value of the bias with which the linear relation is satisfied. In [2] Matsui's method was generalized: rather than keeping one linear relation at each round, the authors kept a list of the best linear approximations and evaluate the value of the biases with the piling-up lemma. However this method is not entirely satisfactory for some reasons: biases that are obtained can be of a much smaller order of magnitude compared to the best bias obtained by Matsui. And to apply previous methods efficiently, it is more important to have relations with biases of the same order as the best biases obtained by Matsui rather than

many relations with much smaller biases. Moreover the method depends heavily on the cipher that is considered. They do not provide a general framework for obtaining the relations.

The main goal of this paper consists in presenting a general purpose algorithm which outputs all linear relations between key, plaintext and ciphertext bits with the best possible biases. More precisely we investigate the problem of finding *all* the linear approximations of a m-variable Boolean function which are satisfied with a given bias ε. As an application, we consider the Boolean functions obtained from the inner product between 8-rounds of the DES and a suitable ciphertext mask. This problem has not yet been addressed in the literature for this kind of cryptanalytic purposes. It can nevertheless be related with the well studied problems of *learning polynomials with queries* in the field of computational learning theory, and of *list-decoding* in the field of coding theory by using methods of maximal likelihood decoding of the first order Reed-Muller codes in a Gaussian channel. That has already been considered for improving fast correlation attacks on stream ciphers, [13]. Different algorithms were designed to solve these problems, see [8, 9, 16, 17]. These algorithms reconstruct the linear relations variable by variable. At every step, a list of the best linear relations is kept and taken as input for the next step. Kabatiansky and Tavernier showed that the maximum size of the list of relations is upper bounded by $1/4\varepsilon^2$ and they proved that the time complexity was upper-bounded by $\mathcal{O}(m^2/\varepsilon^6)$, [16]. If this complexity was close to the complexity on average, then computing multiple linear approximations would become quickly intractable. We propose a significant improvement of the algorithm by Kabatiansky and Tavernier and we apply it to 8 rounds of the DES. We observe experimentally that a complexity of $\mathcal{O}(m/\varepsilon^2)$ is enough to get a list of linear relations with biases of the same order as the best bias obtained by Matsui. Although in the case of the DES proving this average complexity seems difficult, when the Boolean function behaves as if it were the evaluation of a codeword through the binary symmetric channel, this result can be proved from a result by Helleseth, Kløve and Levenshtein, [11].

The paper is organized as follows: in Section 2, we describe the principles on which are based the algorithms searching for linear approximations of a given Boolean function. In a second part we show how to improve the efficiency of Kabatiansky-Tavernier algorithm by performing a complete decoding using the Walsh-Hadamard transform on a number of variables roughly equal to $2\log_2(1/\varepsilon)$, where ε is the expected minimum bias for the relations. In Section 3 we present the results we obtained by running the algorithm on 8-rounds of the DES. We could find more than 80 linear relations on 10 different combinations of ciphertext bits, with biases strictly greater than 1/4-th of Matsui's best bias for 8 rounds of DES. Finally, in Section 4, we present a new method for recovering key bits from the obtained relations. This problem is transformed into the decoding of a first order Reed-Muller code over a Gaussian channel with erasures. We apply this technique to 8 rounds of the DES: we use a list of relations involving 7 information bits for the key, and show how to recover these bits efficiently.

In the rest of the paper, we denote respectively by P, C, K, the plaintext, ciphertext and key vectors of a block cipher, and by "$< \ , \ >$" the usual scalar product of binary vectors.

2 How to find many linear approximations?

The first problem is finding, with significant biases, linear approximations of combinations of ciphertext bits with linear functions of plaintext and key bits.

Given a bias ε, if $|K|$, $|P|$, $|C|$ denote the bit-lengths of the key, plaintext and ciphertext respectively, we want to find the list of all vectors π of length $|P|$, κ of length $|K|$ and γ of length $|C|$, and a bit b, such that

$$< P, \pi > \oplus < K, \kappa > \oplus \, b =< C(P, K), \gamma > \tag{1}$$

is satisfied with probability $\geq 1/2 + \varepsilon$, where the probability is taken over the plaintext and key space.

2.1 Multiple linear approximations and polynomial reconstruction

Let $v \stackrel{def}{=} |P| + |K|$. In equation (1), if we label the bits of the plaintext vector $P = (\delta_0, \ldots, \delta_{|P|-1})$ and the bits of the key vector $K = (\delta_{|P|}, \ldots, \delta_{v-1})$, finding the list of all vectors satisfying equation (1) corresponds to finding the list of all multivariate affine polynomials p of $GF(2)[\delta_0, \ldots, \delta_{v-1}]$ and all the vectors γ of length $|C|$ such that

$$p(\delta_0, \ldots, \delta_{v-1}) =< C(\delta_0, \ldots, \delta_{v-1}), \gamma >$$

is satisfied with probability greater than $1/2 + \varepsilon$.

If the linear combination γ is fixed (we consider this case in the following), then the problem can be considered as a list decoding problem in the first order Reed-Muller code $RM(1, v)$, where the noisy codeword is given by the linear combination of the ciphertext bits γ. This problem comes from the field of computational learning theory, called *learning polynomial with queries* hereafter denoted by LPQ ([24]), which is described in Table 1.

Given: An oracle, the function $f : GF(2)^v \mapsto GF(2)$, the class $\mathcal{C}$ of multivariate affine polynomials in v variables, a parameter ε.
Output: A list of all $p \in \mathcal{C}$ agreeing with f on at least a $1/2 + \varepsilon$ fraction of the inputs.

Table 1. Problem LPQ

Solving this problem is considered to be hard in the general case ([3]). Nevertheless Goldreich and Levin, followed by a generalization of Goldreich, Rubinfeld and Sudan, designed a probabilistic algorithm solving this problem ([8, 9]).

The principle of the algorithm is the following: let $\mathcal{L}$ be the list of affine polynomials in v variables which are solutions to LPQ. Let p be an element of $\mathcal{L}$. The i-prefix of p is by definition the polynomial p limited to the first i variables, that is:

$$p(\delta_0, \ldots, \delta_{i-1}, 0, \ldots, 0).$$

From $i = 0$ to $v - 1$, given a list of candidates $\mathcal{L}_i$ for the i-prefixes of the polynomials in $\mathcal{L}$, the Goldreich-Levin algorithm reconstructs a list of candidates $\mathcal{L}_{i+1}$ for the $i + 1$-prefixes of $\mathcal{L}$ by

1. Adding the $i + 1$-th variable δ_i : $\mathcal{L}_{i+1} = \{s, s + \delta_i \mid s \in \mathcal{L}_i\}$.
2. A screening process eliminating most of the *bad* prefixes candidates.

The efficiency of the algorithm relies directly on the screening process, which should be as optimal as possible.

This algorithm was first modified by Johannson and Jönsson so that it could be adapted to improve fast correlation attacks ([13]). In the case of fast correlation attacks, the queries to the oracle can be considered as random, but cannot be *chosen* randomly, and Johannson and Jönsson designed a specific optimal screening process for that case.

More recently Kabatiansky and Tavernier designed a new deterministic algorithm for list-decoding first order Reed-Muller codes, and showed that it could be transformed into a probabilistic algorithm solving LPQ ([16]). Its complexity was further analysed in [17]. Inspired from the Goldreich-Levin algorithm, it also works by determining $\mathcal{L}$ through the reconstruction of the i-prefixes. A notable difference relies in the screening process which was shown to be optimal in that case ([23]).

The main problem, in trying to apply these algorithms to the search of many linear approximations of a block cipher, is the potential maximum size of the lists $\mathcal{L}_i$. Indeed, from the *Johnson bound* we obtain an upper bound on the list of candidates at every step of $\mathcal{O}(1/\varepsilon^2)$. Moreover this upper bound cannot generally be improved ([16]). Therefore the worst case complexity of the previously mentioned algorithms is

- Memory complexity of order $\mathcal{O}(1/\varepsilon^2)$
- Time complexity at least of $\mathcal{O}(m/\varepsilon^4)$ (see [24]), essentially due to the fact that the list of i-prefixes reaches a size of $\mathcal{O}(1/\varepsilon^2)$ elements.

We note that in the average case, arguments coming from the Helleseth-Klove-Levenshtein paper (see [11]) state that, with high probability, the size of the list is one. We guess that in general our list decoding problem is neither the worst case, nor the average one, but lies between these two extremes.

These evaluations are sufficient to see that finding a list of linear approximations is much more time consuming than the whole linear cryptanalysis whose

complexity is of order $\mathcal{O}(1/\varepsilon^2)$. It is therefore crucial for obvious practical reasons to find a way of avoiding a list of size $\mathcal{O}(1/\varepsilon^2)$. This is the purpose of the next section.

A very recent algorithm ([12]), which is a modified version of the Goldreich-Levin algorithm, has both time and memory complexity of order $\mathcal{O}(m/\varepsilon^2)$. However, it is not appropriate to our cryptographic framework because the memory complexity can not be reduced. In fact, for our problem, we consider that $m = 128$, so avoiding a $\mathcal{O}(m/\varepsilon^2)$ memory complexity is essential.

2.2 Design of the algorithm

To simplify notation in the design and the analysis of the algorithm, we denote by

$$f(\delta_0, \ldots, \delta_{v-1}) \stackrel{def}{=} < C(\delta_0, \ldots, \delta_{v-1}), \gamma >$$

the linear combination of ciphertext bits that we want to approximate within a bias ε.

Our algorithm is based on the principle of the algorithm of Kabatiansky-Tavernier ([17]). This algorithm was chosen because, in our case, its screening process was shown to be optimal [23]. The algorithm obtains the list $\mathcal{L}$ of candidates by reconstructing the prefixes. Let us recall the principle of the screening process. Let $p \in \mathcal{L}$ be an affine polynomial which is solution to the LPQ problem. That means that $d_{\mathcal{H}}(p, f)/2^v \leq (1/2 - \varepsilon)$, where $d_{\mathcal{H}}$ denotes the Hamming distance. Now let $i \leq v$ and let p^i be the i-prefix of p. We have:

$$\begin{aligned} d_{\mathcal{H}}(p, f) &= \sum_{s \in GF(2)^{v-i}} d_{\mathcal{H}}(p(\cdot, s), f(\cdot, s)) \\ &\geq \sum_{s} \min(d_{\mathcal{H}}(p^i, f(\cdot, s)), 2^i - d_{\mathcal{H}}(p^i, f(\cdot, s))). \end{aligned} \tag{2}$$

So an i variable affine polynomial p^i may be the i-prefix of a solution only if the right-hand side quantity of the inequality is less than $2^v(1/2 - \varepsilon)$. In the original algorithm, this quantity is estimated by choosing randomly S vectors $s \in GF(2)^{v-i}$, and for each s, by estimating $d_{\mathcal{H}}(p^i, f(\cdot, s))$ by choosing randomly T vectors $r \in GF(2)^i$.

The main difference with the original algorithm is that we divide it into two steps, which are presented in Table 2 and Table 3:

– The first step is a *decoding step* in the sense that we achieve a *full* decoding: the Hamming distance $d_{\mathcal{H}}(p^\ell, f(\cdot, s))$ is computed exactly for every affine function p^ℓ in ℓ variable, where ℓ is a fixed integer.

We pick up randomly S_1 binary vectors $\mathbf{s}$ of length $v - \ell$, and compute the Walsh-Hadamard transform of the ℓ-variables function $f(\delta_0, \ldots, \delta_{\ell-1}, \mathbf{s})$, that is we compute 2^ℓ dimensional vector

$$\widehat{\mathbf{F}}_{f(\cdot,\mathbf{s})} = (\widehat{F}(0), \ldots, \widehat{F}(2^\ell - 1))$$

where $\widehat{F}$ denotes the Walsh-Hadamard transform of $f(\cdot, \mathbf{s})$, that is

$$\widehat{F}(j) = \sum_{\mathbf{r} \in GF(2^\ell)} (-1)^{f(\mathbf{r},\mathbf{s}) + <\mathbf{r},j>}$$

for j considered as a binary vector of length ℓ by taking its binary form. For $j = 0$ to $2^\ell - 1$, $\widehat{F}(j)$ is an integer between -2^ℓ and 2^ℓ. From $\widehat{F}(j)$, an immediate transform gives the Hamming distance between the ℓ-prefix corresponding to the binary representation of j on ℓ bits and the function $f(\cdot, \mathbf{s})$ (the Hamming distance is $2^{\ell-1} - \widehat{F}(j)/2$). A well known algorithm to compute $\widehat{F}$ can be found in [19]. This is a full decoding since all potential ℓ-prefixes are labelled by their components in the vector $\widehat{\mathbf{F}}_{f(\cdot,\mathbf{s})}$

For every randomly chosen $\mathbf{s}$, the values of $\left|\widehat{\mathbf{F}}_{f(\cdot,\mathbf{s})}\right|$ (the absolute value corresponds to the "min" of equation (2)) are added to the previously obtained values and stored in vector $\mathbf{h}$. Let us denote by $\overline{\mathbf{s}}$ the random variable consisting of a uniform choice among binary vectors of length $v - \ell$. After the S_1 steps, $\mathbf{h}$ is a 2^ℓ dimensional vector and the j-th coordinate h_j contains an average measure of the Hamming distance between the ℓ-prefix corresponding to the integer j and S_1 realisations of the ℓ-variable function $f(\delta_0, \ldots, \delta_{\ell-1}, \overline{\mathbf{s}})$.

The screening process consists in keeping the ℓ-prefixes j such that the average normalized distance between j and the S_1 realisations is less than $1/2 - \varepsilon + \varepsilon/c$, where c is a tolerance parameter: j is a valid prefix if and only if $2^{\ell-1} - (h_j/S_1)/2 \leq 2^\ell(1/2 - \varepsilon + \varepsilon/c)$ (or equivalently if $h_j \geq 2^{\ell+1} S_1 \times \varepsilon(1 - 1/c)$).

– The second part, called *reconstruction step*, follows Kabatiansky-Tavernier algorithm, that was modified to take as input the list of ℓ-prefix candidates issued from the first step of the algorithm. At every step i, the size of the list $\mathcal{L}$ of candidates is first doubled by adding or not adding the variable δ_i to the list of prefixes obtained at step $i - 1$. The screening process is the same as in the *decoding step*. We will choose S_2 binary vector $\mathbf{s}$ of length $v - i - 1$. Given such a vector $\mathbf{s}$, the counter denoted by *Hits* estimates the bias between the i-prefix p and $f(\cdot, \mathbf{s})$ by choosing randomly T vectors of length $i + 1$. And the counter denoted by *Cnt* estimates the average bias between p and S_2 realisation of the function $f(\cdot, \overline{\mathbf{s}})$.

Note that the main difference with the algorithm of Goldreich and Levin is in the screening process. In the latter case, the choice of keeping or rejecting a candidate is not made on average, but if the candidate passes the test for one realisation of $f(\delta_0, \ldots, \delta_{\ell-1}, \overline{\mathbf{s}})$, then it is accepted as a potential ℓ-prefix.

2.3 Analysis of the algorithm

The complexity of the different steps are:

- *Decoding step*: Since the complexity of the Walsh-Hadamard transform on ℓ variables is $\ell 2^\ell$, the complexity of the step is equal to $\mathcal{O}(S_1 \ell 2^\ell)$.
- *Reconstruction step*: The complexity is upper-bounded by $\mathcal{O}((v - \ell) S_2 T L)$, where L denotes the maximum size attained by the list $\mathcal{L}_i$ of i-prefixes through the algorithm.

Decoding Step

– Input:
  - The v variable function f.
  - An estimated bias ε.
  - An integer ℓ giving the number of variables on which the *full* decoding is done.
  - An integer S_1.
  - A tolerance parameter $c > 1$.
– Output: A list $\mathcal{L}$ of linear polynomials in ℓ variable candidates for being ℓ-prefixes.

Algorithm

$\mathcal{L} = \emptyset$
$\mathbf{h} = (h_0 = 0, \ldots, h_{2^\ell - 1} = 0)$,
for $k = 1$ to S_1
  Choose $\mathbf{s} \in GF(2)^{v-\ell}$ randomly
  Compute $\widehat{\mathbf{F}}_{f(\cdot,\mathbf{s})}$
  $\mathbf{h} = (h_0 + \left|\widehat{F}(0)\right|, \ldots, h_{2^\ell-1} + \left|\widehat{F}(2^\ell - 1)\right|)$

for $j = 0$ to $2^{\ell-1}$
  if $h_j \geq 2^{\ell+1} S_1 \times \varepsilon(1 - 1/c)$, **then** $\mathcal{L} = \mathcal{L} \cup \{j\}$
return $\mathcal{L}$

Table 2. Decoding step

Since the size of the list could be as high as $1/4\varepsilon^2$, the question is: what are we gaining by adding a Fourier transform to the algorithm ?

We get the answer in a result of Helleseth, Kløve and Levenshtein on the optimality of the decoding of the first order Reed-Muller codes ([11]). It shows that, if the bias of a function f in v variables is exactly ε, then, provided $2^v > \mathcal{O}(1/\varepsilon^2)$, the size of the list of elements approximating f within ε is reduced to a singleton with a probability of error less than $2^{-2^v\varepsilon^2}$. Therefore if

– ε corresponds to the bias of f,
– the number ℓ of variables on which we do the *decoding step* satisfies $2^\ell > \mathcal{O}(1/\varepsilon^2)$,

then it is sufficient to take $S_1 = \mathcal{O}(1)$ to obtain a list of size $\mathcal{O}(1)$ after the decoding step.

These considerations simply mean that if

– the bias ε is chosen large enough so that the number of linear polynomials approximating f with a probability of $1/2 + \varepsilon$ is constant,
– the integer ℓ on which we perform the Fourier transform is large enough, typically greater than $2\log_2(1/\varepsilon)$,

then the maximum size of the list of candidates in the algorithm will be upper bounded by a constant. In this case it is sufficient to take

Reconstruction step

– Input:
  - A list $\mathcal{L}$ of linear polynomials in ℓ variables.
  - The v variables function f.
  - An estimated bias ε.
  - Integers T and S_2.
  - A tolerance parameter $c > 1$.

– Output: A list $\mathcal{L}$ of linear polynomials in v variables

Algorithm

for $i = \ell$ to $v - 1$
  $\mathcal{L} = \mathcal{L} \cup (\mathcal{L} + \delta_i)$
  for each $p \in \mathcal{L}$
    Cnt $= 0$
    for $k = 1$ to S_2
      Hits $= 0$
      Choose $\mathbf{s} \in GF(2)^{v-i-1}$ randomly
      for $l = 1$ to T
        Choose $\mathbf{r} \in GF(2)^{i+1}$ randomly
        Hits = Hits + $(p(\mathbf{r}) \oplus f(\mathbf{r}, \mathbf{s}))$
      Cnt = Cnt + $\min(T - \text{Hits}, \text{Hits})$
    if Cnt $> TS_2(1/2 - \varepsilon + \frac{\varepsilon}{c})$ then $\mathcal{L} = \mathcal{L} \setminus \{p\}$
return $\mathcal{L}$

Table 3. Reconstruction step

– $S_1 = S_2 = \mathcal{O}(1)$, that is a constant number of trials,
– $T = \mathcal{O}(1/\varepsilon^2)$,

to obtain the list of all linear polynomials approximating f within the given bias.

If we are in a favourable case, the complexity of the algorithm becomes

– Time complexity: $\mathcal{O}\left(\frac{v}{\varepsilon^2}\right)$
– Memory complexity: $\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$

which is an acceptable complexity compared to the complexity of linear cryptanalysis. Namely this work of finding good linear approximations needs do be done only once for each considered cipher.

Note that the so-called tolerance parameter c present in the inputs of the decoding step and reconstruction step corresponds effectively to a tolerance. Namely, since the algorithm is probabilistic, there is a variance around the quantity estimating the bias $1/2 + \varepsilon$, therefore this parameter is necessary to smooth the side effects of the trials.

3 Applications: finding approximations on 8 rounds of DES

Let (P, K) be the concatenate vector (P_H, P_L, K) of length 128 bits in DES (including the eight redundant key bits). Our approach consists in considering the best combination of ciphertext bits $< C(P, K), \gamma >$, as Matsui did, and in giving to the bias the same order of magnitude as the biases found by Matsui. Experience provides us the size of the intermediate lists. It appears that practically, the size of the list is in general rather small (near one hundred). Therefore we can consider that we are in the "favourable" conditions and that the time complexity of the algorithm is $\mathcal{O}(v/\varepsilon^2)$.

For implementing and tuning the algorithm we need to choose the following parameters:

- The number ℓ of variables on which the *decoding step* is achieved.
- The linear combination $f =< C(P, K), \gamma >$ of ciphertext bits.
- The estimated bias ε.
- The tolerance parameter c.

We used the algorithm on different combinations of ciphertext bits of 8-rounds of the DES. The chosen parameters are

- $\varepsilon = 1.8 \times 10^{-4}$. It stands between one third and one fourth of the best bias on 8 rounds given by Matsui: $1.22 \times 2^{-11} = 5.95 \times 10^{-4}$.
- $\ell = 27$: Note that $log(1/\varepsilon^2) \approx 25 \leq 27$, as indicated in previous section.
- The number of samples $S_1 = 20 = \mathcal{O}(1)$ and $S_2 = 16 = \mathcal{O}(1)$. This gives experimentally good choices.
- The number $T = 1/\varepsilon^2 = 2^{25}$
- The tolerance parameter c was set to 1.9 after some experiments. Indeed if it is too big, the size of the list can be empty and if it is too small, the size of the list can be huge.

With these choices, at every step of the reconstruction procedure, the list of candidates does not exceed experimentally 100 and the running time of the algorithm on a Pentium 4 at 3.00GHz is approximately of 1 day.

Because of the lack of space, we were not able to put all the obtained equations in this paper, but some of the most significant ones are presented in Table 4 and Table 5. The notation for the equations is the same as Matsui's one, meaning that P_H, P_L (resp. C_H, C_L) correspond to the left and right 32 bits of plaintext (resp. ciphertext). One of the main differences is that for simplicity of the algorithm, all the key bits are theoretically involved in finding the linear approximations rather than the round keys.

In Table 4 we used the combination $< C(P, K), \gamma >$ corresponding to Matsui's combination of ciphertext bits given in the original paper [20]. We obtained 10 linear approximations with the same order of magnitude as the bias given by Matsui.

In Table 5, we used as linear combination of ciphertext bits the best linear combination of plaintext bits from Table 4. Thanks to the symmetry of the DES, we found again Matsui's linear combination.

To obtain other linear combinations with significant biases, and because of symmetries of the DES, we took as the linear combination of ciphertext bits some of the linear combinations of plaintext bits with significant biases obtained through the previous steps of the algorithm. This method enabled us to obtain more than 80 linear combinations on 10 different linear combinations of ciphertext bits. However note that the same bits appear in many equations therefore a system consisting of all the obtained equations is not of full rank. This could be a drawback when applying the techniques of linear cryptanalysis using many multiple linear approximations since these approximations are not linearly independent.

Bias	Linear Combination		
5.79×10^{-4}	$P_L[12,16]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,15,21,23,25,27,42,54,57,60]$
3.73×10^{-4}	$P_L[15]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,15,21,23,25,36,54,57,60]$
3.40×10^{-4}	$P_L[11,14]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,15,21,23,54,57]$
-2.73×10^{-4}	$P_L[11,13,16]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,10,15,21,23,25,42,54,57]$
2.67×10^{-4}	$P_L[11,12,13,16]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,10,15,21,23,25,27,42,54,57]$
2.30×10^{-4}	$P_L[12,15]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,15,21,23,25,27,36,54,57,60]$
-2.19×10^{-4}	$P_L[13,14,16]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,10,15,21,23,42,54,57,60]$
-2.13×10^{-4}	$P_L[13,14]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,10,15,21,23,54,57,60]$
2.03×10^{-4}	$P_L[11,12,13]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,10,15,21,23,25,27,54,57]$
-1.72×10^{-4}	$P_L[11,13,14]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,10,15,21,23,54,57]$
2.44×10^{-4}	$P_L[14]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,15,21,23,54,57,60]$
-3.51×10^{-4}	$P_L[14,16]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,15,21,23,42,54,57,60]$
0.95×10^{-4}	$P_L[13,14,15,17]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,10,15,21,23,42,54,57,60]$
2.33×10^{-4}	$P_L[13,16]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,10,15,21,23,25,42,54,57]$
0.82×10^{-4}	$P_L[11,13,15,16]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,10,15,21,23,25,36,42,54,57]$
0.84×10^{-4}	$P_L[13,15]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,10,15,21,23,25,36,54,57,60]$
-3.31×10^{-4}	$P_L[15]$	$\oplus\ P_H[7,18,24,29]$	$\oplus\ K[4,14,15,21,23,25,36,54,57,60]$
-0.81×10^{-4}	$P_L[11,12,14,16]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,15,21,23,27,42,54,57]$
2.06×10^{-4}	$P_L[12,14]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,15,21,23,27,54,57,60]$
3.91×10^{-4}	$P_L[12,14,16]$	$\oplus\ P_H[7,18,24]$	$\oplus\ K[4,15,21,23,27,42,54,57,60]$

Table 4. Ciphertext bits combination: $C_H[15] \oplus C_L[7,18,24,27,28,29,30,31]$

4 Soft decoding to reconstruct the key

We propose in this part a soft decoding technique to reconstruct the key. We want to establish a correspondence between linear relations found with the decoding algorithm (as those given by Table 5) and a codeword y from the first order Reed-Muller code which is generated by the master key and which is noisy by a Gaussian channel with erasures. Let us denote $P = (P_H, P_L)$ and $C = (C_H, C_L)$. We are in the case where we have the following relations:

$$\langle C, \gamma_i \rangle + \langle P, \pi_i \rangle + \langle K, \kappa_i \rangle = 0 \tag{3}$$

Bias	Linear Combination	
-2.49×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,31]$	$\oplus K[4,9,13,31,33,41,44,52,54]$
4.86×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,27,31]$	$\oplus K[4,9,13,31,33,41,44,47,52,54]$
-4.68×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,28]$	$\oplus K[4,9,15,31,33,41,44,52,54]$
4.81×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,27,28]$	$\oplus K[4,9,15,31,33,41,44,47,52,54]$
-2.18×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,27,28,29,31]$	$\oplus K[9,13,15,31,33,41,44,47,52,54]$
-3.67×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,27,28,31]$	$\oplus K[4,9,13,15,31,33,41,44,47,52,54]$
-4.59×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,30]$	$\oplus K[4,9,30,31,33,41,44,52,54]$
2.63×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,27,30]$	$\oplus K[4,9,30,31,33,41,44,47,52,54]$
2.3×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,29,30,31]$	$\oplus K[9,13,30,31,33,41,44,52,54]$
2.69×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,27,29,30,31]$	$\oplus K[9,13,30,31,33,41,44,47,52,54]$
3.77×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,30,31]$	$\oplus K[4,9,13,30,31,33,41,44,52,54]$
3.23×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,27,30,31]$	$\oplus K[4,9,13,30,31,33,41,44,47,52,54]$
2.43×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,27,28,29,30]$	$\oplus K[9,15,30,31,33,41,44,47,52,54]$
-3.33×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,28,30]$	$\oplus K[4,9,15,30,31,33,41,44,52,54]$
-3.13×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,28,29,30,31]$	$\oplus K[9,13,15,30,31,33,41,44,52,54]$
4.52×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,28,30,31]$	$\oplus K[4,9,13,15,30,31,33,41,44,52,54]$
2.05×10^{-4}	$P_H[15] \oplus P_L[7,18,24,27,31]$	$\oplus K[4,9,13,31,33,41,44,47,52]$
2.48×10^{-4}	$P_H[15] \oplus P_L[7,18,24,27,28,30,31]$	$\oplus K[4,9,13,15,30,31,33,41,44,47,52]$
4.82×10^{-4}	$P_H[15] \oplus P_L[7,18,24,31]$	$\oplus K[4,9,13,31,33,41,44,52]$
2.05×10^{-4}	$P_H[15] \oplus P_L[7,18,24,27,31]$	$\oplus K[4,9,13,31,33,41,44,47,52]$
2.49×10^{-4}	$P_H[15] \oplus P_L[7,18,24,28,29,31]$	$\oplus K[9,13,15,31,33,41,44,52]$
-3.4×10^{-4}	$P_H[15] \oplus P_L[7,18,24,27,28,31]$	$\oplus K[4,9,13,15,31,33,41,44,47,52]$
3.55×10^{-4}	$P_H[15] \oplus P_L[7,18,24,29,30]$	$\oplus K[9,30,31,33,41,44,52]$
-2.31×10^{-4}	$P_H[15] \oplus P_L[7,18,24,27,30]$	$\oplus K[4,9,30,31,33,41,44,47,52]$
2.28×10^{-4}	$P_H[15] \oplus P_L[7,18,24,27,28,29,30]$	$\oplus K[9,15,30,31,33,41,44,47,52]$
5.83×10^{-4}	$P_H[15] \oplus P_L[7,18,24,27,28,29,30,31]$	$\oplus K[9,13,15,30,31,33,41,44,47,52]$
-2.57×10^{-4}	$P_H[15] \oplus P_L[7,18,24,28,30,31]$	$\oplus K[4,9,13,15,30,31,33,41,44,52]$
1.06×10^{-4}	$P_H[15] \oplus P_L[7,18,24]$	$\oplus K[4,9,31,33,41,44,52]$
-1.17×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,27,28,29]$	$\oplus K[9,15,31,33,41,44,47,52,54]$
-1.09×10^{-4}	$P_H[15] \oplus P_L[7,18,24,30,31]$	$\oplus K[4,9,13,30,31,33,41,44,52]$
1.16×10^{-4}	$P_H[15] \oplus P_L[0,7,18,24,27,28,29,30,31]$	$\oplus K[9,13,15,30,31,33,41,44,47,52,54]$

Table 5. Ciphertext bits combination: $C_L[12,16] \oplus C_H[7,18,24]$

which hold with probability $p_i = 1/2+\varepsilon_i$, for $i = 1,\ldots,k$. Let $\Delta_0, \Delta_1, \ldots, \Delta_t$ be vectors such that the set of key-bits involved in the previous relations belongs to the t-dimensional affine space $\Delta_0 + Vect(\Delta_1,\ldots,\Delta_t)$. For $X = (X_1,\ldots,X_t) \in \{0,1\}^t$, we use the notation $\Delta \cdot X \stackrel{def}{=} \Delta_0 + \Delta_1 X_1 + \cdots + \Delta_t X_t$.

To mount the attack, we consider that we have a sample of size s of plaintext-ciphertext pairs, associated with the key $\bar{K}$. Our aim is to determine, using (3), the affine function $A(X) \in RM(1,t)$ defined by

$$A(X) = \langle \bar{K}, \Delta \cdot X \rangle.$$

This affine function generates precisely the codeword that we have to reconstruct, $A(X)$ gives the value of this codeword at position X. This will lead to the knowledge of the (linear combination of) key bits $K[\Delta_i]$, $i = 0,\ldots,t$. The idea is to construct a noisy and erased codeword y which is close enough to the codeword A, to be able to decode it in the first order Reed-Muller code $RM(1,t)$.

Using the mapping $\alpha \in \{0,1\} \mapsto (-1)^\alpha$, we consider that the codeword y belongs to $\mathbb{R}^n$. Then, the most probable codeword $a \in RM(1,t)$ (for A) is given by the one that leads to the maximum (among all codewords) inner product $\sum_{x\in\{0,1\}^t}(-1)^{a(x)}y(x)$ with the received vector y.

For $i = 1, \ldots, k$, let $x_i \in \{0,1\}^t$ be the vector corresponding to κ_i, such that $\kappa_i = \Delta \cdot x_i$. We construct y as follows. For $i = 1, \ldots, k$, let s_1 be the number (among the sample) of terms $\langle C, \gamma_i \rangle + \langle P, \pi_i \rangle$ equal to 1 and s_0 be the number of such terms equal to 0. Here $s_1 + s_0 = s$. Now let $P_1 = (1/2 - \varepsilon_i)^{s_0} \times (1/2 + \varepsilon_i)^{s_1}$ and $P_0 = (1/2 - \varepsilon_i)^{s_1} \times (1/2 + \varepsilon_i)^{s_0}$. Then the probability that $A(x_i) = 1$ equals $p_1 = P_1/(P_0 + P_1)$, and the probability that $A(x_i) = 0$ equals $p_0 = P_0/(P_0 + P_1)$. Then we will choose for the value of y at position x_i the soft-quantity $\ln(p_1/p_0)$:

$$y(x_i) = s_0 \ln\left(\frac{1/2 - \varepsilon_i}{1/2 + \varepsilon_i}\right) + s_1 \ln\left(\frac{1/2 + \varepsilon_i}{1/2 - \varepsilon_i}\right).$$

Manipulating log-probability quantities, rather than working with the probabilities themselves, is generally preferred due to computational issues such as a finite-precision representation of numbers, and since the log-probability quantities represent information as it is defined in the field of Information Theory. Soft information yields reliability measures for the received bits and is generated from channel observations in the physical layer.

The positions X for which we have not any approximation (i.e. for $X \in \{0,1\}^t \setminus \cup_i x_i$) will be vanished (by setting $y(X) = 0$), because we can consider this position as erasures. It is well known that, on average, the first order Reed-Muller can be efficiently decoded in a Gaussian channel and erasure channel, and we refer to the result of I. Dumer and R. Krichevskiy (see [6]) for more details concerning the performances of this code in a Gaussian channel. Note that, given a set of relations, the dimension t has to be "small" so that the number of erased position is reasonable.

Application to 8 rounds of DES

We apply here the soft decoding technique using the relations of Table 5. The key-bits involved in these relations belong to an affine subspace of dimension 6, defined by $\Delta_0 = K[9, 31, 33, 41, 44, 52]$, $\Delta_1 = K[4]$, $\Delta_2 = K[13]$, $\Delta_3 = K[15]$, $\Delta_4 = K[30]$, $\Delta_5 = K[47]$ and $\Delta_6 = K[54]$. We show in Fig. 1 the (experimental) success rate of the attack, depending on the size of the sample (between 500000 and 2000000). We also show the success rate of the attack if we set $y(x_i) = \pm 1$ instead of the log-probability (1 if $p_1 > p_0$, -1 otherwise).

5 Conclusion

The algorithm we designed enabled us to find many multiple linear approximations of 8 rounds of DES. These approximations can be used in improving the efficiency of linear cryptanalysis. Though the obtained relations are not all linearly independent, this first attempt is enough to improve quite significantly the data complexity of linear cryptanalysis on 8 rounds of the DES. The problem of finding more ciphertext masks, leading to linear approximations with good biases, remains open.

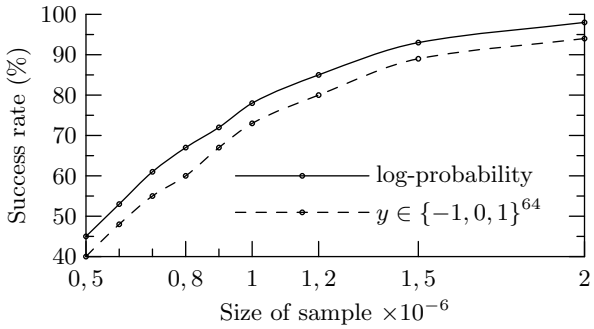

Fig. 1. Experimental success rate of the attack.

We have also proposed an original algorithm based on soft decoding that permits to reconstruct efficiently key bits. We did this attack completely for the set of equations of Table 5. The second set of equations of Table 4 suggests us that a similar attack could provide at least ten extra key bits information. We also tried to find approximations on 16 rounds of DES, by chaining relations. For example, using Matsui's best 12-rounds relation and decoding the 4 remaining rounds gives 26 suitable relations for the attack, involving 6 potentially reconstructed key-bits. However we have not yet evaluated the corresponding success rate.

Acknowledgement

We gratefully thank Professor Ilya Dumer for very helpful discussions about soft decision decoding.

References

1. A. Biryukov, C. De Cannière, and M. Quisquater. On multiple linear approximations. In M. Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3512 of *Lecture Notes in Computer Science*, pages 1–22. Springer, 2004.
2. A. Biryukov, C. De Cannière, and M. Quisquater. On multiple linear approximations. Cryptology ePrint Archive 2004/057, 2004.
3. A. Blum, M. Furst, M. Kearns, and R. Lipton. Cryptographic primitives based on hard learning problems. In *Advances in Cryptology – CRYPTO'93*, volume 773 of *Lecture Notes in Computer Science*, pages 278–291. Springer, 1993.
4. B. Gérard Le Bobinnec. Utilisation de codage correcteur d'erreurs pour la cryptanalyse de systèmes de chiffrement à clé secrète. Master's thesis, Université de Versailles, September 2007.
5. B. Collard, F. X. Standaert, and J.-J. Quisquater. Experiments on the multiple linear cryptanalysis of serpent. In *Fast Software Encryption, FSE 2008*, 2008.
6. I. Dumer and R. Krichevskiy. Soft decision majority decoding of Reed-Muller codes. *IEEE Transactions on Information Theory*, 46(1):258–264, 2000.

7. B. Gérard and J.-P. Tillich. On linear cryptanalysis with many linear approximations. Technical report, 2007. preprint.
8. O. Goldreich and L. A. Levin. A hard core predicate for all one-way functions. In *Proceedings of the 21-st ACM Symposium on Theory of Computing*, pages 25–32, May 1989.
9. O. Goldreich, R. Rubinfeld, and M. Sudan. Learning polynomials with queries: the highly noisy case. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 294–303, 1995. Extended version: http://people.csail.mit.edu/madhu/papers.html.
10. C. Harpes, G. G. Kramer, and J. L. Massey. A generalization of linear cryptanalysis and the applicability of matsui-s piling up lemma. In L. Guillou and J. J. Quisquater, editors, *Advances in Cryptology – EUROCRYPT'95*, volume 921 of *Lecture Notes in Computer Science*, pages 24–38. Springer, 1995.
11. T. Helleseth, T. Kløve, and V. Levenshtein. Bounds on the error-correcting capability of codes beyond half the minimum distance. In D. Augot, P. Charpin, and G. Kabatianski, editors, *Proceedings of the 3rd International Workshop on Coding and Cryptography, WCC 2003*, pages 243–251, 2003.
12. G. Kabatiansky I. Dumer and C. Tavernier. The Goldreich-Levin algorithm with reduced complexity. Technical report, 2007. preprint.
13. T. Johansson and F. Jönsson. Fast correlation attacks through reconstruction of linear polynomials. In M. Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 300–315. Springer, 2000.
14. P. Junod. On the complexity of matsui's attack. In S. Vaudenay and A. M. Youssef, editors, *Selected Areas in Cryptography (SAC'01)*, volume 2259, pages 199–211. Springer, 2001.
15. P. Junod. On the optimality of linear differential and sequential distinguishers. In *Advances in Cryptology - EUROCRYPT'03*, 2003.
16. G. Kabatiansky and C. Tavernier. List decoding of Reed-Muller codes. In *Ninth International Workshop on Algebraic and Combinatorial Coding Theory, ACCT'2004*, pages 230–235, June 2004. http://ced.tavernier.free.fr/Balgaria.pdf.
17. G. Kabatiansky and C. Tavernier. List decoding of first order Reed-Muller codes II. In *Tenth International Workshop on Algebraic and Combinatorial Coding Theory, ACCT'2006*, pages 131–134, September 2006. http://ced.tavernier.free.fr/Kabat.pdf.
18. B. Kaliski and M. Robshaw. Linear cryptanalysis using multiple linear approximations. In Y. Desmedt, editor, *Advances in Cryptology – CRYPTO'94*, Lecture Notes in Computer Science, pages 26–39. Springer, 1994.
19. F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error–Correcting Codes*. North Holland, 1977.
20. M. Matsui. Linear cryptanalysis method for the DES cipher. In *Advanced in cryptology - EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer, 1993.
21. M. Matsui. The first experimental cryptanalysis of the Data Encryption Standard. In Y. Desmedt, editor, *Advances in Cryptology – CRYPTO'94*, Lecture Notes in Computer Science, pages 1–11. Springer, 1994.
22. S. Murphy. The independance of linear approximations in symmetric cryptanalysis. *IEEE Transactions on Information Theory*, 52(12):5510–5518, December 2006.
23. C. Tavernier. *Testeurs, problmes de reconstruction univariés et multivariés et application à la cryptanalyse du DES*. PhD thesis, Ecole Polytechnique, 2004.
24. L. Trevisan. Some applications of coding theory in computational complexity. In *Electronic Colloquium on Computational Complexity*, number 43, 2004.