Vectorial Fast Correlation Attacks[*]

Jovan Dj. Golić
Access Network and Terminals
Telecom Italia Lab, Telecom Italia
Via G. Reiss Romoli 274, 10148 Turin, Italy
Guglielmo Morgari
Telsy Elettronica e Telecomunicazioni
Corso Svizzera 185, 10149 Turin, Italy

Abstract

A new, vectorial approach to fast correlation attacks on binary memoryless combiners is proposed. Instead of individual input sequences or their linear combinations, the new attack is targeting subsets of input sequences as a whole, thus exploiting the full correlation between the chosen subset and the output sequence. In particular, all the input sequences can be targeted simultaneously. The attack is based on a novel iterative probabilistic algorithm which is also applicable to general memoryless combiners over finite fields or finite rings. Experimental results obtained for randomly chosen binary combiners with balanced combining functions show that the vectorial approach yields a considerable improvement in comparison with the classical, scalar approach.

Key words Vectorial correlation attack, sequential linear cryptanalysis, iterative probabilistic decoding, memoryless combiners.

1 Introduction

Fast correlation attacks on binary linear feedback shift registers (LFSR's) in keystream generators for stream cipher applications are important cryptanalytic techniques which are introduced in [9] and [13], and are based on an earlier work [12] on divide-and-conquer correlation attacks. The attacks exploit the bitwise correlation between the keystream sequence

[*]The first version of this paper entitled "Vectorial approach to fast correlation attacks," by Jovan Dj. Golić and Philip Hawkes, was submitted to Crypto 2002 and was accepted for *Designs, Codes and Cryptography*, to be published in 2005 (see http://www.kluweronline.com/issn/0925-1022). The new version contains more detailed algorithms and a different experimental setting. Both versions are independent of the paper "A new cryptanalytic attack on PN-generators filtered by a Boolean function," by S. Leveiller, G. Zemmor, P. Guillot, and J. Boutros, in Selected Areas in Cryptography - SAC 2002, *Lecture Notes in Computer Science*, vol. 2595, pp. 232-249, 2003.

and linear combinations of the LFSR sequences. As such, they are directly applicable to memoryless combiners, but can be also extended to combiners with memory. A binary memoryless combiner is a well-known type of keystream generators which consists of a number of LFSR's whose output sequences are bitwise transformed by a nonlinear combining function into the keystream sequence. The correlation can be represented by a binary symmetric channel whose capacity is typically very small. The goal is to reconstruct the combined LFSR sequence from an observed segment of the keystream sequence in the known-plaintext scenario. The problem is equivalent to one of decoding a truncated cyclic linear (N, k) block code, where k is the combined LFSR length and N is the observed keystream segment length. In this problem, k is large and the rate k/N is very small in order for the correlation attack to be successful. Consequently, the optimum decoding minimizing the block-error rate (e.g., the minimum distance decoding [12] for a time-invariant binary symmetric channel) is not feasible.

The decoding techniques used in fast correlation attacks are based on the linear relations satisfied by the codeword bits which are called the parity checks. The parity checks correspond to polynomial multiples of the combined LFSR feedback polynomial. For the attacks to be effective, the parity checks should have a relatively low weight, i.e., should involve a small number of codeword bits. The techniques essentially reduce to (iterative) error-correction decoding algorithms for binary symmetric channels. They may be feasible for large k and $N - k$. For example, an iterative hard-decision decoding technique [13] is based on the majority decision rule and originates from [3], where a similar technique is based on a more sophisticated iteration principle, later called the belief propagation principle. An iterative soft-decision decoding technique first proposed in [9] and later improved in [10] essentially originates from [1] and [3], and is based on the posterior probability symbol-by-symbol decoding introduced in [8] for orthogonal parity checks. In addition, there have been more recent contributions in this area such as [7], [11], and [5].

Naturally, techniques based on soft-decision decoding are more effective than techniques based on hard-decision decoding, whereas iterative decoding techniques are more powerful than one-step decoding techniques. The required output sequence length and the complexity of such attacks mainly depend on the absolute value(s) of the exploited correlation coefficient(s) and on the degrees and numbers of low-weight polynomial multiples of the involved LFSR feedback polynomials.

One motivation for this paper is to generalize the binary fast correlation attacks to q-ary fast correlation attacks that can be applied to q-ary memoryless combiners, in which the LFSR sequences are defined over a finite field $\mathbf{F}_q$ or over a finite ring of integers $\mathbf{Z}_q$. Such attacks are important for analyzing the keystream generators suitable for software applications, which typically utilize linear recurrencess involving a small number of terms over $\mathbf{F}_q$ or $\mathbf{Z}_q$, where $q = 2^m$. However, our main objective to show that 2^m-ary fast correlation attacks can be applied to binary memoryless combiners, such as nonlinear filter generators, thus simultaneously utilizing the correlations to different linear functions of m inputs in a vectorial manner. Experimental results obtained by computer simulations demonstrate that the vectorial attacks are more powerful than the ordinary binary (scalar) correlation attacks.

A probabilistic model for the sequential linear cryptanalysis over finite fields and the basic one-step algorithm for updating the underlying probability distributions are introduced in Section 2. The corresponding iterative probabilistic algorithms for fast correlation attacks over finite fields $\mathbf{F}_{2^m}$ are developed in Section 3. Their vectorial application to binary memoryless combiners is explained in Section 4. Experimental results comparing the vectorial and scalar fast correlation attacks on binary memoryless combiners are presented in Section 5, and conclusions are given in Section 6.

2 Sequential Linear Cryptanalysis over Finite Fields

In this section, a probabilistic model for fast correlation attacks over finite fields is identified and a symbol-by-symbol approach for solving the underlying problem is proposed. The approach can also deal with linear congruential relations instead of linear relations over finite fields.

2.1 Probabilistic Problem

Let $\mathbf{X} = (X_i)_{i=1}^N$ be a sequence of independent variables over a finite field $\mathbf{F}_q$ with the prior probability distributions $\Pr\{X_i = x\} = P_i(x)$, $x \in \mathbf{F}_q$, $1 \le i \le N$. Let L denote an $(N-k)$-tuple of linearly independent linear functions defined on $\mathbf{F}_q^N$, let $\mathbf{Y} = L(\mathbf{X})$, and let $L^{-1}(\mathbf{0}) = \{\mathbf{X} \in \mathbf{F}_q^N | L(\mathbf{X}) = \mathbf{0}\}$, where $|L^{-1}(\mathbf{0})| = q^k$. Then the posterior probability distribution of $\mathbf{X}$ conditioned on the event that the linear relations among the variables induced by L are satisfied is for any $\mathbf{X} \in L^{-1}(\mathbf{0})$ given as

$$\Pr\{\mathbf{X} \mid L(\mathbf{X}) = \mathbf{0}\} \;=\; \frac{\Pr\{\mathbf{X}\}}{\Pr\{L(\mathbf{X}) = \mathbf{0}\}} \;=\; \frac{\prod_{i=1}^N \Pr\{X_i\}}{\Pr\{L(\mathbf{X}) = \mathbf{0}\}} \tag{1}$$

where $\Pr\{L(\mathbf{X}) = \mathbf{0}\} = \sum_{\mathbf{X} \in L^{-1}(\mathbf{0})} \prod_{i=1}^N \Pr\{X_i\}$. Our objective is to find the most likely solution to the system of linear equations $L(\mathbf{X}) = \mathbf{0}$, that is, an $\mathbf{X} \in L^{-1}(\mathbf{0})$ that maximizes this conditional probability. However, q^k steps are required to find such a solution, and this is infeasible if q^k is large. The problem is related to one of decoding the linear code (vector subspace) $L^{-1}(\mathbf{0})$ where instead of specifying the communication channel, the prior probability distributions of individual variables are given directly.

2.2 Symbol-by-Symbol Approach

Another approach would be to find a solution that maximizes each of the posterior probability distributions for individual variables in $\mathbf{X}$ when conditioned on $L(\mathbf{X}) = \mathbf{0}$, that is, $\Pr\{X_i = x | L(\mathbf{X}) = \mathbf{0}\}$, $x \in \mathbf{F}_q$, $1 \le i \le N$. Then only qN steps are required, provided that these probability distributions are already computed. Their exact computation can be achieved in q^{N-k} steps by adapting the Hartmann-Rudolph algorithm [6]. The computation utilizes the set $\mathcal{L}$ of all q^{N-k} linear relations, called parity checks, satisfied by every $\mathbf{X} \in L^{-1}(\mathbf{0})$, namely, all the linear combinations of $N-k$ linear functions in L. For linear codes, this algorithm minimizes the decoding error probability for individual symbols rather than for blocks of symbols. This computation is infeasible if q^{N-k} is large, but in certain cases approximations can be effective. For example, it would be interesting to investigate if it is possible to extend the approximation to the Hartmann-Rudolph algorithm developed for $q = 2$ in [5].

In this paper, we provide an approximation that generalizes a well-known expression used for bit-by-bit probabilistic decoding of binary linear codes based on orthogonal parity checks (e.g., see [8], [3], and [1]). That expression is used in binary fast correlation attacks (e.g., see [9] and [10]). Let L_i denote a set of linear relations from $\mathcal{L}$ involving the i-th variable X_i, for each $1 \le i \le N$. Each relation should preferrably have a low *weight*, defined as the number of involved variables minus 1. It is also desirable, but not necessary, that the relations in each L_i be orthogonal on X_i, i.e., that X_i be the only variable they share in common. Let a generic linear relation $l \in L_i$ of weight w be put into the form

$$X_i \;=\; \sum_{j=1}^{w} a_j X_{i_j} \;\overset{\text{def}}{=}\; X_i^l \tag{2}$$

where $a_j \neq 0$, $1 \le j \le w$.

The approximation $\hat{P}_i(x)$ for $\Pr\{X_i = x | L(\mathbf{X}) = \mathbf{0}\}$ can then be obtained in two stages. First, iteratively compute the probability distribution of every variable X_i^l by using the convolution expression for the probability distribution of the sum of two independent variables X and Y over $\mathbf{F}_q$

$$\Pr\{X + Y = z\} \;=\; \sum_{x \in \mathbf{F}_q} \Pr\{X = x\} \Pr\{Y = z - x\}, \quad z \in \mathbf{F}_q. \tag{3}$$

Second, for each $1 \le i \le N$, in view of

$$\Pr\{X_i = x | L_i(\mathbf{X}) = \mathbf{0}\} \;=\; \Pr\{X_i = x\} \frac{\Pr\{L_i(\mathbf{X}) = \mathbf{0} | X_i = x\}}{\Pr\{L_i(\mathbf{X}) = \mathbf{0}\}}, \tag{4}$$

compute

$$\hat{P}_i(x) \;=\; P_i(x) \frac{\prod_{l \in L_i} \Pr\{X_i^l = x\}}{\sum_{y \in \mathbf{F}_q} P_i(y) \prod_{l \in L_i} \Pr\{X_i^l = y\}}, \quad x \in \mathbf{F}_q. \tag{5}$$

The expression (5) is obtained by using the fact that the variables X_i^l, $l \in L_i$, are mutually independent if the linear relations in L_i are orthogonal on X_i, but can also be used if they are not orthogonal. Note that this expression is not exact even if the linear relations in L_i are orthogonal, because it does not make use of their linear combinations.

The convolution of a number of probability distributions tends to be uniform if this number increases, and uniform distributions effectively do not contribute to (5). This is why it is important that the weight of the employed linear relations be not too high.

2.3 Complexity

In order to compute (5) for every $1 \le i \le N$, it is required to compute the probability distribution of X_i^l for every used linear relation l and for every X_i involved in l. If l has weight

w, then the direct application of (3) would require the computation of $w+1$ convolutions of w probability distributions, i.e., altogether w^2-1 convolutions of two probability distributions. However, it is simple to see that the same can be achieved by computing only $3(w-1)$ convolutions of two probability distributions. The convolution itself takes $(q-1)q$ real multiplications when computed by (3).

The complexity can be reduced by using the Fourier transform of the probability distributions. The transform can be defined for any $\mathbf{F}_q$ (or $\mathbf{Z}_q$), but, for simplicity, assume that $q = 2^m$. Using a vectorial representation of the field elements, let $x = (x_1, \cdots, x_m)$ and $\omega = (\omega_1, \cdots, \omega_m)$. A generic linear function of x parametrized by ω, $\mathbf{F}_2^m \to \mathbf{F}_2$, can be expressed as the inner product $\omega \cdot x = \sum_{j=1}^m \omega_j x_j \bmod 2 = \omega_1 x_1 \oplus \cdots \oplus \omega_m x_m$. Then a probability distribution P and its Fourier transform $\mathbf{P}$ are related by

$$\mathbf{P}(\omega) \; = \sum_{x \in \mathbf{F}_q} P(x)(-1)^{\omega \cdot x}, \qquad P(x) \; = \; \frac{1}{q} \sum_{\omega \in \mathbf{F}_q} \mathbf{P}(\omega)(-1)^{\omega \cdot x}. \tag{6}$$

Alternatively, the generic linear function can also be expressed as $\mathrm{Tr}(\omega x)$, where $\mathrm{Tr} : \mathbf{F}_q \to \mathbf{F}_2$ ($\mathrm{Tr}(\alpha) = \alpha + \alpha^2 + \alpha^{2^2} \cdots + \alpha^{2^{m-1}}$) is the trace function with respect to $\mathbf{F}_2$. The two Fourier transforms are equivalent up to an invertible linear function of ω and reduce to the well-known Walsh-Hadamard transform. Both transforms can be computed by a fast Fourier transform algorithm in $O(q \log_2 q)$ steps.

As the Fourier transform of the convolution of two probability distributions is the product of their Fourier transforms, the convolution of two probability distributions can thus be computed in $O(q \log_2 q)$ instead of $O(q^2)$ steps. In addition, with respect to the trace function representation, if a variable with a probability distribution P is multiplied by a constant a, then the Fourier transform for the new variable is simply $\mathbf{P}(a\omega)$. If $a \in \mathbf{F}_2$, that is, if the multiplication is componentwise, $ax = (ax_1, \cdots, ax_m)$, then the same holds for the Fourier transform (6), with respect to the inner product representation. The latter is then more convenient than the former if the multiplicative constants in the linear relations (2) all belong to the ground field $\mathbf{F}_2$.

Let λ denote the average number of the used linear relations per variable and let w_{av} denote their average weight. The total complexity of computing the posterior probability distributions of all N variables is then $O(w_{\mathrm{av}} \lambda N q \log_2 q)$. The required space is $O(Nq)$.

2.4 Correlation Coefficients

The Walsh-Hadamard transform of a probability distribution P as defined by (6) can be interpreted in terms of the correlation coefficients of linear functions. Namely, $\mathbf{P}(\omega)$ is the expected value of $(-1)^{\omega \cdot X}$ with respect to the probability distribution P which itself is equal to the correlation coefficient between the linear function $\omega \cdot X$ and the constant zero if the vector X is randomly chosen according to P, that is,

$$\mathbf{P}(\omega) = c(\omega) \; = \; \Pr\{\omega \cdot X = 0\} - \Pr\{\omega \cdot X = 1\}. \tag{7}$$

Thus, $\mathbf{P}(\omega)$ completely specifies the probability distribution of $\omega \cdot X$ and is in fact equal to the Walsh-Hadamard transform coefficient of this distribution. The inverse Walsh-Hadamard transform is then simply determined by

$$\Pr\{\omega \cdot X = 0\} \;=\; \frac{1}{2}\,(1 + c(\omega)). \tag{8}$$

The minimal absolute value, zero, of $\mathbf{P}(\omega)$ is achieved if and only if P is such that $\omega \cdot X$ is uniformly distributed (balanced), and its maximal absolute value, 1, is achieved if and only if P is such that $\omega \cdot X$ has a single value with probability 1. In particular, $\mathbf{P}(0) = 1$. Accordingly, $|\mathbf{P}(\omega)|$ is a measure of nonuniformity of the probability distribution of $\omega \cdot X$ and $\operatorname{sign} \mathbf{P}(\omega)$ indicates the more likely value of $\omega \cdot X$, which enables one to make hard decisions on $\omega \cdot X$. More generally, in view of the orthogonality of the Walsh-Hadamard transform, the index

$$I \;=\; \sum_x P^2(x) \;=\; \frac{1}{q}\,(1 + \sum_{\omega \neq 0} |\mathbf{P}(\omega)|^2) \tag{9}$$

is a measure of nonuniformity of P. Its minimal value, $1/q$, is achieved if and only if P is uniform, and its maximal value, 1, is achieved if and only if there exists a single value x such that $P(x) = 1$.

3 Iterative Probabilistic Algorithms

In this section, several types of iterative probabilistic algorithms based on the symbol-by-symbol update of the probability distributions are presented. Typically, we assume that $q = 2^m$, so that the symbols are then represented as binary vectors.

3.1 Vectorial Recycling

Instead of computing the posterior probability distributions of individual variables only once, we can proceed iteratively, in each iteration substituting the computed posterior probability distributions for the prior probability distributions in the next iteration. Apart from this *direct recycling*, we can also use the recycling based on the belief propagation principle by generalizing the binary approach from [3] (see also [2] and [5]). The iterations are useful because (5) is only an approximate expression and because the hard decisions, maximizing the posterior probability distributions of individual variables, generally do not result in sequences belonging to $L^{-1}(\mathbf{0})$, i.e., satisfying all the linear relations from L. However, in the recycling, the denominator in (5) may become equal to zero, which means that it is not possible to satisfy all the linear relations from L_i for each value of X_i. For each such i, $\hat{P}_i(x)$ is then reset to the initial probability distribution $P_i(x)$.

To enhance the interaction between different variables during the iterations, the expression (5) can be recycled in such a way that explicitly present terms for the prior probability distributions are kept at their initial values in every iteration, so that the probability distributions are updated through the linear combination variables X_i^l only. This is called *modified direct recycling.* In this way, we also overcome the problem present in the direct recycling that if $\hat{P}_i(x)$ (wrongly) becomes equal to zero in some iteration, for some i and some x, then it remains equal to zero in all the subsequent iterations.

Experimental results indicate that the vectorial recycling algorithm typically converges after a certain number of iterations, and, in the case of direct recycling, the limit probability distributions are the fixed points of the underlying nonlinear operator. The modified direct recycling is generally slower, but results in the limit probability distributions closer to a vectorial sequence from $L^{-1}(\mathbf{0})$. Note that any vectorial sequence can be represented as a sequence of the probability distributions each of which has value 1 for a single vectorial value. In the final stage, vectorial hard decisions can be taken according to the limit probability distributions. In the case of success, the obtained sequence $\bar{\mathbf{X}}$ will be at a small Hamming distance from a sequence from $L^{-1}(\mathbf{0})$, which can then be recovered by a q-ary information set decoding algorithm. More precisely, in the error-free information set decoding approach, one randomly chooses a subset of k variables (positions) in $\bar{\mathbf{X}}$ (possibly with a relatively high nonuniformity index I), assumes that they are free of errors, and then recovers $\mathbf{X}$ by solving the system of linear equations $L(\mathbf{X}) = \mathbf{0}$. The unique solution will exist if and only if the chosen variables are linearly independent when restricted to $L^{-1}(\mathbf{0})$, that is, if and only if there are no linear relations in $\mathcal{L}$ involving only the chosen variables. Among the found candidate solutions, the one with the largest $\prod_{i=1}^{N} \Pr\{X_i\}$ is identified as the most likely solution.

However, it is likely that the limit probability distributions do not result in a vectorial sequence close to a sequence from $L^{-1}(\mathbf{0})$, with respect to the Hamming distance, and yet the algorithm can be made successful because of the obtained limit probability distributions being closer to a sequence from $L^{-1}(\mathbf{0})$ than the initial probability distributions. This can possibly be achieved by applying the scalar recycling and/or resetting algorithms described in Sections 3.2 and 3.3, respectively. They are especially interesting if the linear functions from L involve only (bitwise) multiplication by constants from $\mathbf{F}_2$. Then, for $q = 2^m$, each linear function from L can be decomposed into m identical linear functions over the binary components of q-ary variables and the corresponding binary linear relations can be used for recovering the component binary sequences.

In this case, it may also suffice to apply a binary information set decoding algorithm directly to the $2^m - 1$ linear combination binary sequences $\omega \cdot \mathbf{X} = (\omega \cdot X_i)_{i=1}^{N}$, for nonzero ω. Let Q_i, $1 \leq i \leq N$, be the limit probability distributions of the vectorial recycling algorithm. We first find all i and all ω such that $|c_i(\omega) = \mathbf{Q}_i(\omega)| \geq a$, for some high threshold such as $a = 0.9$. Then we produce a system of linear equations by making hard decisions on the linear functions $\omega \cdot X_i$ according to the rule: if $c_i(\omega) \geq 0$, then $\omega \cdot X_i = 0$, and if $c_i(\omega) < 0$, then $\omega \cdot X_i = 1$. The error-free information set decoding is then applied to this system.

3.2 Scalar Recycling

Starting from some initial probability distributions Q_i, $1 \leq i \leq N$, possibly obtained as the limit of the vectorial recycling algorithm, the objective of the scalar recycling algorithm is to recover some of the $2^m - 1$ binary sequences $\omega \cdot \mathbf{X}$, for nonzero ω. Each of the binary sequences is treated individually by using the binary version of the vectorial recycling algorithm. More precisely, the scalar recycling algorithm starts from the correlation (Walsh-Hadamard transform) coefficients $c_i(\omega)$ and then, for each ω separately, keeps updating these correlation coefficients, or the corresponding probabilities, by using the binary ($q = 2$) versions of (2), (3), and (5). For each ω, the scalar (binary) recycling algorithm is actually the same as iterative probabilistic algorithm used in binary fast correlation attacks (e.g., see [5]). The convergence of the binary algorithm is much faster than if q is relatively large.

In the case of success for a given ω, the corresponding binary sequence satisfying all the linear relations from L can then be recovered by taking hard decisions and by applying a low-complexity binary information set decoding algorithm. If the algorithm is successful for at least m linearly independent values of ω, then the q-ary sequence is recovered directly by solving the corresponding linear equations. If the algorithm is successful for at least one value of ω, then the initial probability distributions can be recomputed and the whole attack (vectorial and scalar) repeated, with more chances on success, to recover other $\omega \cdot \mathbf{X}$ and so on until the whole vectorial sequence $\mathbf{X}$ is reconstructed.

3.3 Resetting

Let Q_i, $1 \leq i \leq N$, be the probability distributions obtained in some iteration of the vectorial recycling algorithm. The resetting algorithm is applied to their Walsh-Hadamard transforms in such a way that the signs are preserved while the absolute values are reset to the values corresponding to the initial probability distributions. Namely, $\mathbf{Q}_i(\omega)$ is modified into $\operatorname{sign} \mathbf{Q}_i(\omega) \cdot |\mathbf{P}_i(\omega)|$. If $\mathbf{Q}_i(\omega) = 0$, then the sign of $\mathbf{P}_i(\omega)$ is taken instead.

A justification for this definition of resetting is that the information about the signs of the correlation coefficients, or, equivalently, about the hard decisions on the linear combination sequences $\omega \cdot \mathbf{X}$ is much more important than the absolute values themselves. In particular, suppose that all the signs are correct, that is, $\operatorname{sign} \mathbf{Q}_i(\omega) = (-1)^{\omega \cdot X_i^*}$, for some $\mathbf{X}^* \in L^{-1}(\mathbf{0})$. In other words, suppose that the binary hard decisions directly yield a solution. Then, regardless of the absolute values, one can theoretically show that the binary (direct) recycling algorithm converges to $\omega \cdot X_i^*$ (e.g., see [5]). We also experimentally found that the vectorial recycling algorithm always converged to $\mathbf{X}^*$, which shows that the algorithm is sound.

The resetting algorithm can be applied to the limit probability distributions of the vectorial recycling algorithm, before starting the scalar recycling algorithm. Experiments show that better results can be obtained by the so-called fast resetting when the resetting is periodically performed after a specified number of iterations during the vectorial recycling algorithm. The same holds for the scalar recycling algorithm.

3.4 Stopping Criteria

It is natural to expect that the updated probability distributions become more and more concentrated with every iteration of the vectorial recycling algorithm. This is confirmed by experiments which show that the nonuniformity index I (9), averaged over the sequence length, increases with every iteration until the limit is reached, especially so if the algorithm is successful. So, the stopping criterion for the vectorial recycling algorithm is when I

reaches 1 or when the ratio of the values of I in the current and the preceding iteration drops below a threshold close to 1, such as 1.001, which was used in the experiments. If this ratio gets smaller than 1, then the output probability distributions are taken from the preceding iteration. With fast resetting, I is computed after each update and after each resetting, but the algorithm can stop only after the update of probability distributions, not after resetting. A specified maximal number of iterations is needed as an additional stopping criterion, especially with fast resetting, when the convergence is slower.

For the scalar recycling algorithm, the convergence is typically very fast, so that the nonuniformity index stopping criterion is not very useful. If this algorithm is applied after the vectorial recycling algorithm, then only the final stopping criterion is needed. For the version with fast resetting, this is either a reached maximal number of iterations (failure) or when, for at least one ω, the estimated correlation coefficient between the linear combination sequence reconstructed by hard decisions and a binary sequence satisfying all the linear relations from L, is sufficiently high, say at least 0.9 (estimated success). More precisely, the algorithm is stopped if for 10 iterations the number of estimated successes is not changed. For each ω, the correlation coefficient estimate can be obtained from the total number of satisfied linear relations. For example, if each used linear relation involves exactly 3 terms, then the estimate can be obtained as the cubic root of the relative difference between the total number of satisfied and unsatisfied linear relations.

4 Application to Memoryless Combiners

A memoryless combiner is a common type of keystream generators for stream ciphers which consists of a number of LFSR's whose output sequences are symbol-wise transformed by a nonlinear combining function into the keystream sequence. For practical reasons, we will restrict ourselves to binary combiners, in which the LFSR sequences are defined over $\mathbf{F}_2$.

One can then use the bitwise correlation between the output sequence and linear combinations of input LFSR sequences in order to reconstruct the initial states of the involved LFSR's by applying the well-known binary (fast) correlation attacks introduced in [12], [9], and [13]. An iterative probabilistic algorithm for this scalar attack is explained in Section 3.2. The correlation coefficients between a combining n-bit Boolean function f and all linear n-bit functions can be computed in $n2^n$ steps by the fast Walsh-Hadamard transform algorithm. Namely, use (6) with $q = 2^n$ and $(-1)^f/2^n$ instead of P, to obtain $\mathcal{F}(\omega) = 2^{-n}\sum_{x\in\mathbf{F}_2^n}(-1)^{f(x)+\omega\cdot x}$. Here $\mathcal{F}(\omega)$ is the correlation coefficient between $f(x)$ and $\omega \cdot x$. The required output sequence length and the complexity of such an attack mainly depend on the absolute value(s) of the exploited correlation coefficient(s) and on the degrees and numbers of low-weight polynomial multiples of the involved LFSR feedback polynomials which define the parity checks. (The weight of a polynomial is the number of its nonzero coefficients.)

The iterative probabilistic algorithms introduced in Section 3 enable us to introduce another, more general and more effective approach. In this, so-called vectorial approach, we utilize the symbol-wise correlation between the output binary sequence and a subset of m input binary LFSR sequences as a whole. This subset is an m-dimensional vectorial binary sequence, which satisfies the linear recurrence defined by the least common multiple, h, of the involved LFSR feedback polynomials. Therefore, the vectorial sequence is a 2^m-ary sequence satisfying a binary linear recurrence, so that the probabilistic model from Section 2.1 is applicable. The prior probability distributions of the corresponding 2^m-ary random variables are then defined as the conditional probability distributions $\Pr\{X' = x \,|\, f(X) = y\}$, where X' is the corresponding m-dimensional subvector of the n-dimensional binary vector X, and the output values y are obtained from a given output sequence. If $m = n$, then we have

$$P(x|y) = \Pr\{X = x \mid f(X) = y\} \;=\; \frac{1}{|\mathcal{X}_y|}, \quad x \in \mathcal{X}_y = \{x \in \mathbf{F}_2^n | f(x) = y\}, \tag{10}$$

whose Walsh-Hadamard transform is given as $\mathbf{P}(\omega|y) = \sum_{x\in\mathcal{X}_y}(-1)^{\omega\cdot x}/|\mathcal{X}_y|$. For $\omega \neq 0$, we further have $\mathbf{P}(\omega|y) = (-1)^y(2^{n-1}/|\mathcal{X}_y|)\mathcal{F}(\omega)$. Note that $\mathcal{F}(0) = 2^{-n}(|\mathcal{X}_0| - |\mathcal{X}_1|)$.

Accordingly, given a known segment $(y_i)_{i=1}^N$ of the output sequence, the corresponding prior probability distributions are $P_i(x) = P(x|y_i)$, and their Walsh-Hadamard transforms are then $\mathbf{P}_i(\omega) = \mathbf{P}(\omega|y_i)$. In the first iteration of the vectorial recycling algorithm, the posterior probability distributions $\hat{P}_i(x)$ are then given by (5), where, for a generic linear relation l of the form $X_i = \sum_{j=1}^w X_{i_j}$, the probability distribution of $X_i^l = \sum_{j=1}^w X_{i_j}$ is determined by

$$\Pr\{X_i^l = x\} \;=\; \frac{1}{2^n}\left(1 + (-1)^{w_1}\frac{2^{w(n-1)}}{|\mathcal{X}_0|^{w-w_1}|\mathcal{X}_1|^{w_1}}\sum_{\omega\neq 0}\mathcal{F}(\omega)^w(-1)^{\omega\cdot x}\right), \tag{11}$$

where w_1 denotes the Hamming weight of $(y_{i_1}, \cdots, y_{i_w})$. If f is balanced, then $|\mathcal{X}_0| = |\mathcal{X}_1| = 2^{n-1}$ and $\mathcal{F}(0) = 0$, so that (11) reduces to

$$\Pr\{X_i^l = x\} \;=\; \frac{1}{2^n}\left(1 + (-1)^{y_{i_1}\oplus\cdots\oplus y_{i_w}}\sum_{\omega\in\mathbf{F}_2^n}\mathcal{F}(\omega)^w(-1)^{\omega\cdot x}\right). \tag{12}$$

The vectorial approach is more powerful than the scalar approach, because it simultaneously makes use of the total correlation between the output sequence and a chosen subset of input LFSR sequences instead of using the correlation to individual linear combinations of these sequences separately. This means that for a given segment of the output sequence, the vectorial correlation attack may be able to reconstruct the initial states of the input LFSR's in the cases when the scalar correlation attack is not successful. The required output sequence length and the complexity of the vectorial attack depend on the used conditional probability distributions and on the degrees and numbers of low-weight polynomial multiples of h, similarly as for the scalar attack.

The most interesting case is when $m = n$, that is, when the target of the attack are all the input LFSR sequences combined. In this case, the vectorial correlation attack exploits the full correlation between the binary output and the n-dimensional binary input to a given binary combining function f. For balanced f, the nonuniformity index of the initial probability distributions is then equal to $2^{-(n-1)}$. Given n, the success of the attack is therefore expected to be less dependent on f itself. Both vectorial and scalar attacks are especially effective if all the input LFSR's have the same feedback polynomial, preferably of low weight, because the least common multiple polynomial h is then always equal to this feedback polynomial, regardless of the subset of inputs chosen. Nonlinear filter generators represent a common type of such memoryless combiners.

5 Experimental Results

The objective of the experiments performed by computer simulations was to check the convergence properties of the vectorial recycling algorithm (VRA) from Section 3.1 and to compare the performance of an iterative probabilistic algorithm based on the VRA with the performance of the scalar recycling algorithm (SRA) from Section 3.2.

For comparison purposes, the experiments were conducted on binary memoryless combiners consisting of a variable number, n, of LFSR's with the same feedback polynomial of degree r and of fixed weight 3. We used balanced combining functions f with a controllable range of the maximal absolute value, $|c|_{\max}$, of the correlation coefficients to linear functions. The used parity checks, of weight 2, were obtained by repeatedly squaring the LFSR feedback polynomial. The output sequence length was of the form $N = r2^j$, in which case the average number of parity checks per input vector is given as $\lambda_j = 3(j - 1 + 2^{-j})$. The performance of the attack for a fixed f then predominantly depends on the parameter j, and not on r. Accordingly, to maximize the number of experiments, as the time and space complexities of the considered algorithms linearly increase with r, we picked $r = 20$. Also, note that choosing a larger weight of the LFSR feedback polynomial would require a larger N, whereas choosing different LFSR feedback polynomials would generally require a much larger N, because the parity checks should then be obtained from their least common multiple, which is unlikely to have weight 3 (e.g., by using the method from [4]).

The chosen vectorial attack uses the total correlation to all the input LFSR sequences combined. It consists of running the VRA until convergence and then of running the SRA. The scalar attack consists of running the SRA only. Both VRA and SRA use periodic fast resetting after every 3 iterations, and the resetting is also applied after the VRA before starting the SRA. The VRA is based on the modified direct recycling, while the SRA uses the direct recycling. The maximal number of iterations in both VRA and SRA was set to 99 in the experiments. For each n, each j, and a given range of $|c|_{\max}$ (low, medium, and high, according to Table 1), the experiment consists first of randomly choosing the combining function and the LFSR initial states and then of running the vectorial and scalar attacks separately. The number of experiments was 100 for $n = 4, 5, 6$ and 50 for $n = 7$. *In this setting, the advantage to be expected from the vectorial attack over the scalar attack is in the improved initial probability distributions for the SRA.* In fact, in many cases it also happens that the VRA itself already recovers the original vectorial input sequence.

In each experiment and for each ω, the measure of success is the true correlation coefficient between the linear combination sequence $\omega \cdot \mathbf{X}$ reconstructed by hard decisions and the linear

n	4	5	6	7
$2^n\|c\|_{\max}$	≤ 8, ≥ 12	≤ 8, $[12,20]$, ≥ 24	≤ 12, $[16,36]$, ≥ 40	≤ 20, $[24,48]$, ≥ 52

Table 1: Ranges of maximal absolute values of correlation coefficients.

combination sequence $\omega \cdot \mathbf{X}^*$, where $\mathbf{X}^*$ is the original vectorial input sequence. Recall that the estimated correlation coefficient, computed from the number of satisfied parity checks, is used for stopping the SRA. If for some ω both the correlation coefficients are at least 0.9, we then say that a solution is found, because $\omega \cdot \mathbf{X}^*$ can then easily be reconstructed by a simple error-free information set decoding algorithm. For each experiment, the number of obtained solutions is recorded and if this number is 1 or more, then the experiment is considered to be successful. The more the solutions, the easier the further reconstruction, and if the number of the solutions, for linearly independent ω, is n or larger, then the whole input sequence can immediately be reconstructed. For a nonlinear filter generator, as the LFSR sequences are phase shifts of each other, only one solution is enough. The main obtained results are summarized in Fig. 1 and Fig. 2.

Fig. 1 displays the success rates of the vectorial and scalar attacks as functions of the average number of parity checks per bit, λ_j, for $n = 5, 6, 7$ and for the low and medium ranges of $|c|_{\max}$. Similar results are obtained for $n = 4$. There is a considerable improvement achieved by the vectorial attacks. The main advantage is to be expected for the low range of $|c|_{\max}$, because the SRA is then less likely to be successful, while the VRA can be successful as it exploits the combined correlation to all the linear combinations of the input sequences simultaneously. For the high range of $|c|_{\max}$, the difference between the vectorial and scalar attacks is less significant as both the attacks approach the 100% success rate faster. We also performed a number of successful vectorial attacks for $n = 8$, for the low range of $|c|_{\max}$ ($2^n|c|_{\max} \leq 36$), and for $j \geq 7$ ($N \geq 2560$).

In order to increase the improvement for larger n, instead of a simple periodic fast resetting with period 3, an adaptive fast resetting can be utilized for the VRA. Other optimizations of the vectorial attack including intertwined vectorial and scalar recycling algorithms may also be possible. An interesting observation regarding the SRA is that in many cases it was successful only after a relatively large number of iterations, namely, 50 or more, due to the periodic fast resetting. In any case, the fast resetting improved the performance of both the VRA and SRA.

Fig. 2 shows the average number of solutions obtained by the vectorial attack as a function of λ_j, for $n = 5, 6, 7$ and for low, medium, and high ranges of $|c|_{\max}$. There is a significant improvement achieved by the vectorial attacks, because for the scalar attacks, this number is only 1 or very close to 1 in all the cases. We observed that the number of solutions may depend on the chosen parameters for the VRA, for example, on the period of the fast resetting.

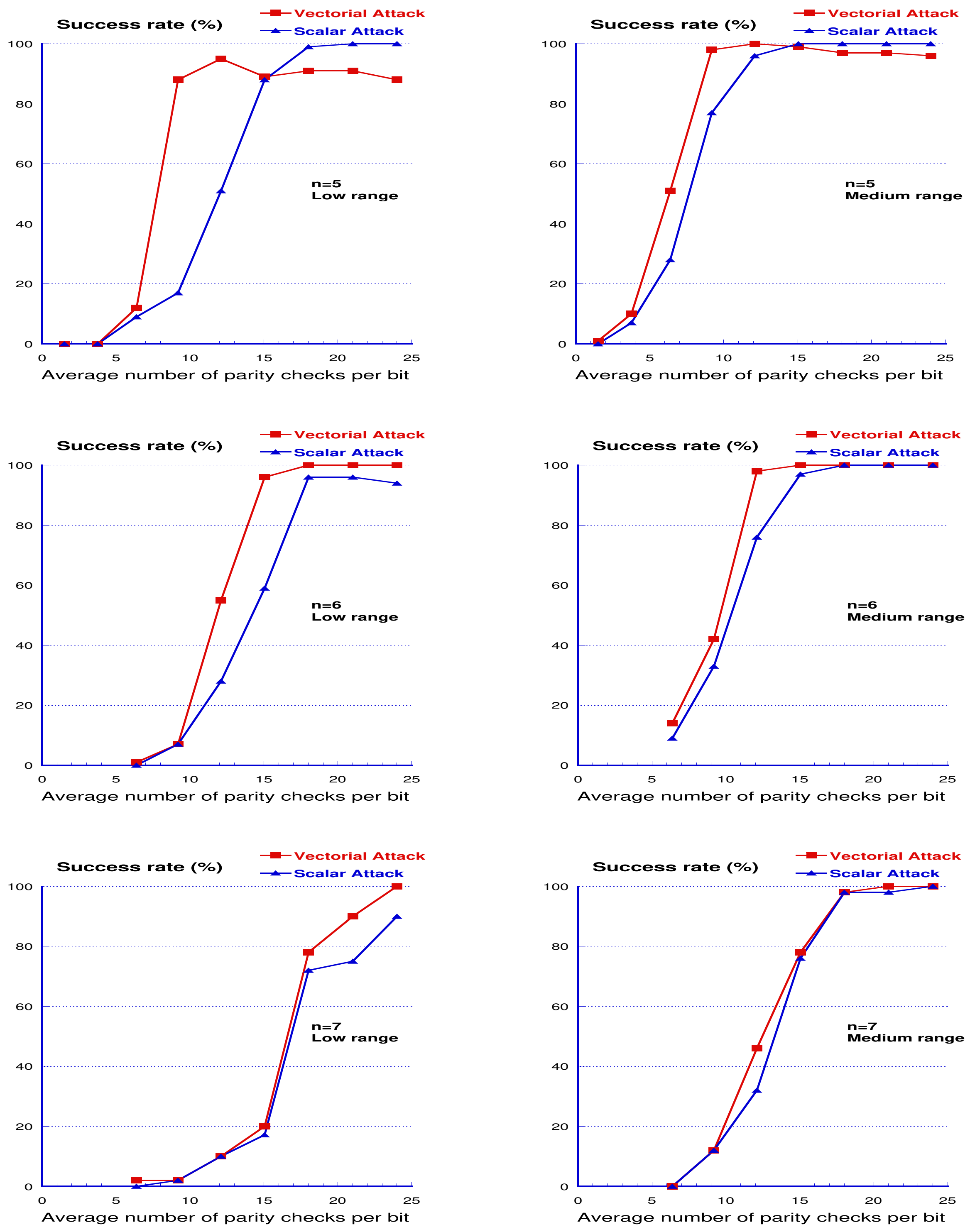
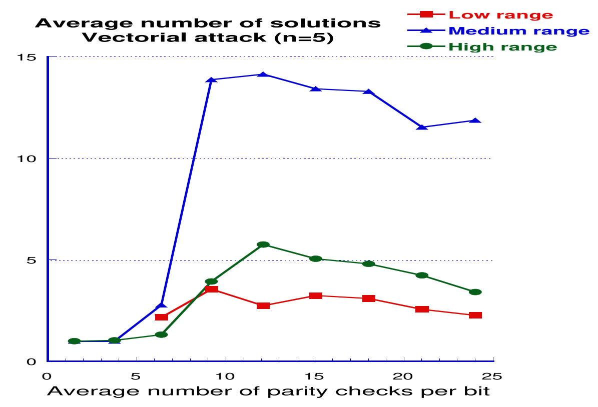
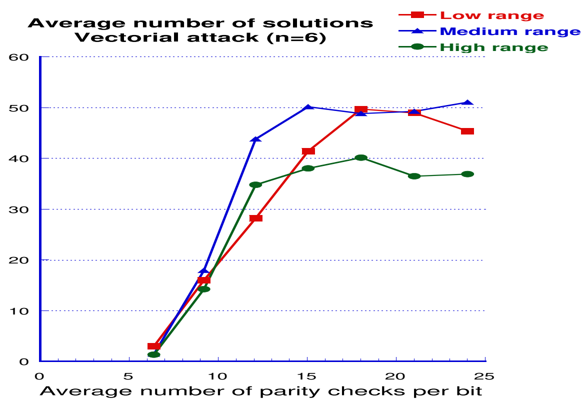
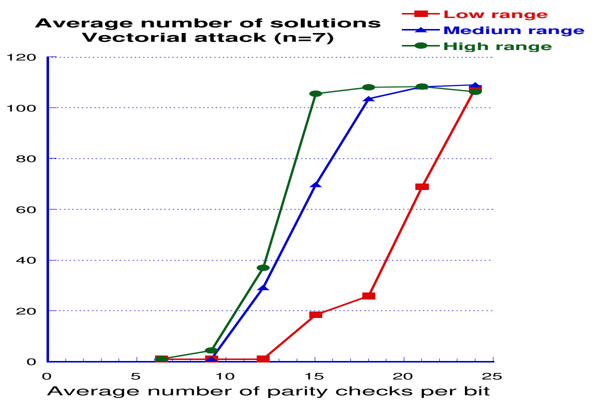


Figure 1: Success rates for vectorial and scalar attacks for low and medium ranges of the correlation coefficient, for $n = 5, 6, 7$.







Average number of solutions
Scalar attack

Range	Low	Medium	High
n=5	1.20	1.34	1.01
n=6	1.15	1.38	1.02
n=7	1.05	1.26	1.21

Figure 2: Average number of solutions for vectorial attack in case of success, for $n = 5, 6, 7$. For scalar attack, only the average value over j is presented, as it is very close to 1.

6 Conclusion

The developed vectorial fast correlation attacks are more powerful than the classical scalar fast correlation attacks because they make use of the total correlation between the output sequence and a targeted subset of input LFSR sequences in a memoryless combiner. The attacks are based on a novel iterative probabilistic algorithm which utilizes the Fourier transforms of the underlying conditional probability distributions. The new attacks, when applied to all the input sequences simultaneously, are less dependent on the combining function than the classical attacks. Experiments show that, for a given set of parity checks used, the vectorial approach can be successful when the scalar approach is not as well as that the number of reconstructed linear combinations of input sequences is significantly larger. In particular, the vectorial attack can be successful even for very short output sequences. Further algorithmic optimizations of the vectorial attack are possible. Another problem interesting for future investigations is a theoretical analysis of the conditions for its successful convergence, but is expected to be very difficult.

The vectorial fast correlation attacks are also applicable to combiners over arbitrary finite fields or finite rings of integers. These combiners are suitable for software applications and typically involve linear recurrences containing a small number of terms which makes the attacks more effective.

References

[1] G. C. Clark, Jr., and J. B. Cain, *Error-Correcting Coding for Digital Communications*, Plenum Press, New York, 1982.

[2] M. P. C. Fossorier, M. J. Mihaljević, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Trans. Commun.*, vol. 47, pp. 673-680, May 1999.

[3] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. 8, pp. 21-28, Jan. 1962.

[4] J. Dj. Golić, "Computation of low-weight parity-check polynomials," *Electronics Letters*, vol. 32, pp. 1981-1982, Oct. 1996.

[5] J. Dj. Golić, "Iterative optimum symbol-by-symbol decoding and fast correlation attacks," *IEEE Trans. Inform. Theory*, vol. 47, pp. 3040-3049, Nov. 2001.

[6] C. R. P. Hartmann and L. D. Rudolph, "An optimum symbol-by-symbol decoding rule for linear codes," *IEEE Trans. Inform. Theory*, vol. 22, pp. 514-517, Sept. 1976.

[7] T. Johansson and F. Jonnson, "Improved fast correlation attacks on stream ciphers via convolutional codes," Advances in Cryptology - EUROCRYPT '99, *Lecture Notes in Computer Science*, vol. 1592, pp. 347-362, 1999.

[8] J. L. Massey, *Threshold Decoding*, MIT Press, Cambridge, MA, 1963.

[9] W. Meier and O. Staffelbach, "Fast correlation attacks on certain stream ciphers," *Journal of Cryptology*, vol. 1, pp. 159-176, 1989.

[10] M. J. Mihaljević and J. Dj. Golić, "A comparison of cryptanalytic principles based on iterative error-correction," Advances in Cryptology - EUROCRYPT '91, *Lecture Notes in Computer Science*, vol. 547, pp. 527-531, 1991.

[11] M. J. Mihaljević, M. P. C. Fossorier, and H. Imai, "A low-complexity and high-performance algorithm for the fast correlation attack," Fast Software Encryption - FSE 2000, *Lecture Notes in Computer Science*, vol. 1978, pp. 196-212, 2001.

[12] T. Siegenthaler, "Decrypting a class of stream ciphers using ciphertext only," *IEEE Trans. Comput.*, vol. 34, pp. 81-85, Jan. 1985.

[13] K. Zeng and M. Huang, "On the linear syndrome method in cryptanalysis," Advances in Cryptology - CRYPTO '88, *Lecture Notes in Computer Science*, vol. 403, pp. 469-478, 1990.