Classification Based on Euclidean Distance Distribution for Blind Identification of Error Correcting Codes in Noncooperative Contexts

Aurélien Bonvard, Sébastien Houcke, Roland Gautier, Mélanie Marazin

Classification Based on Euclidean Distance Distribution for Blind Identification of Error Correcting Codes in Non-Cooperative Contexts

Aurélien Bonvard, *Member, IEEE,* Sébastien Houcke, *Member, IEEE,* Roland Gautier, *Member, IEEE,* and Mélanie Marazin, *Member, IEEE,*

***Abstract*—The use of channel code is mandatory in current digital communication systems. It allows to access the information on the receiver side despite the presence of noise. In this paper, we are interested in the blind identification of the parameters of an error correcting code from a received noisy data stream. Literature provides a large amount of contributions for this problem in the hard-decision case but few in the soft-decision case. It is well known that soft-decision methods allow significant gain in decoding techniques. Thence, we propose an algorithm which is able to identify the length of a code through a classification process from the bits likelihood values. It highlights a difference of behavior between an i.i.d. sequence and an encoded one. This method does not rely on any a priori knowledge about the encoder involved. Indeed, the distribution of n-length code words in an n-dimensional space depends on the encoder characteristics. Some areas of this n-dimensional space are left vacant because of the redundancy added by the encoder. Despite the presence of noise, it is still possible to detect this phenomenon. Furthermore, an adaptation of a collisions method based on the birthday paradox gives us access to an estimation of the code dimension. Finally, we investigate the performance of this estimation methods to show their efficiency.**

***Index Terms*—Blind identification, Coding, Electronic Warfare, Interception, Non-Cooperative Context.**

I. Introduction

In current telecommunication systems, the use of error correcting codes allows to retrieve information by correcting errors due to a noisy channel. To perform such a correction, redundant information is added before sending it through the channel. On the receiver side, after the demodulation and synchronization processes, the sent sequences are decoded and some (or all) errors are corrected.

In traditional schemes, transmission parameters are known by the transmitter and the receiver. In non-cooperative context, such as electronic warfare, a third party needs to blindly reconstruct the emitted message from the received signal without knowing the coding parameters. In AMC, we may have partial information available (i.e. list of potential code used). The proposed method could be used in any of these cases with little adaptations. It estimates the code length and rate. We assume here that the acquired signal has been successfully demodulated and frame synchronized. Therefore, we consider a BPSK-modulated stream at the output of an additive white Gaussian noise (AWGN) channel. In this context, the problem considered here is formulated as follows: "given this received signal, how do we identify the parameters of the involved encoder?". We aim to blindly determine the parameters of a binary linear block code: namely, its length and its dimension. In this paper, we extend some concepts of the hard-decision context to the soft-decision context. For instance, we adapt the collision counting principle proposed in [1]: we count an amount of classes instead. Each class can be viewed as colliding elements in an n-dimensional space.

Over the last few years, many methods using hard bits information to identify the channel coding have been proposed for linear codes [2]–[9], LDPC codes [5], [6], convolutional codes [1], [8], [10]–[15], turbo-codes [16], [17], Reed-Solomon [18] and cyclic codes [19]. More recently, new methods based on soft bits information have been proposed [20], [21]. A Log Likelihood Ratio's (LLR) metrics analysis is performed to identify an encoder from a *candidate set* in the context of the Adaptive Modulation and Coding (AMC). These algorithms establish the likelihood of each candidate code and choose the most likely. The LLR computation is obtained from the syndrome posterior probability (i.e. the probability that all the parity check relations of the code are satisfied). Furthermore, some techniques such as [22] allows the receiver to reconstruct the sub-coder of a turbo code. However, Yu *et al.* assume the sub-code parameters to be known (length, dimension and constraint length). The blind identification of the encoder parameters is a prerequisite to blind reconstruction. The main objective of this article is to propose an answer to this need.

In the context of electronic warfare, a catalogue of possible candidate codes may not be available and the encoder parameters remain the first elements to identify. It is well known that soft-decision methods allow significant gain in decoding techniques. Thence, we propose an algorithm which is able to identify the length of a code through a classification process from the bits likelihood values. The noisy intercepted stream is divided into contiguous blocks of equal length. They are then regrouped into classes. This classification is operated with a Euclidean distance criterion. A threshold is established, if the distance between two blocks is lower than this threshold, they belong to the same class. When the block size reaches the code length, the number of classes drops: namely the *number*

All the authors are with the Lab-STICC Laboratory (UMR CNRS 6285)

A. Bonvard and S. Houcke are with the Department of Signal and Communications, IMT Atlantique, Technopôle Brest-Iroise, CS 83818, 29238 Brest cedex 03, France. e-mail: aurelien.bonvard@imt-atlantique.fr and sebastien.houcke@imt-atlantique.fr

R. Gautier and M. Marazin are with the Universty of Brest, 6 avenue Le Gorgeu, 29238 Brest, France. e-mail: roland.gautier@univ-brest.fr and melanie.marazin@univ-brest.fr

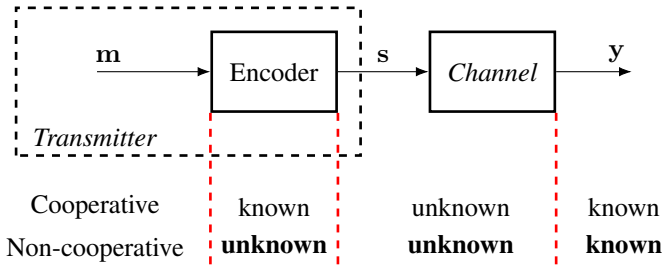
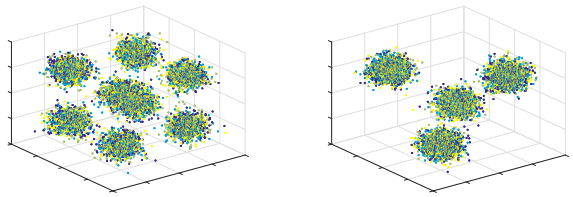


Fig. 1. Comparison of cooperative and non-cooperative contexts



(a) (b)

Fig. 2. Observation of noisy words in 3-dimensional Euclidean space: (a) without encoding, (b) with encoding $n_c = 3$ and $k_c = 2$.

of classes deficiency principle. We define blocks sharing a class as colliding blocks. Also, the amount of collisions is related to the code dimension. From this, we manage to estimate the code dimension. Indeed, with the proper distance threshold the noisy words distribution is related to a classic *birthday problem*.

In the following, we give both formal and intuitive definitions of the number of classes deficiency. Section II is related to the code length estimation. We present the general aspects of the classification method and a detailed description of the associated algorithm. We also give specifications concerning the threshold on the Euclidean distances. Section III deals with the code dimension estimation. Finally, to confirm the interest in this new algorithm, some results are discussed in section IV.

A. Model and notations

Without loss of generality, we consider $\mathcal{C}(n_c, k_c)$, a binary linear encoder which length and dimension are respectively denoted n_c and k_c with $k_c < n_c$ on the transmitter side. The transmission model is described by the Figure 1. In the non-cooperative context, the intercepted stream $\mathbf{y}$ is the only available knowledge about the transmission from the eavesdropper point of view.

On the receiver side, we consider we access a reliability measure from each intercepted bit: a LLR. We therefore consider that the received data result from a BPSK modulation and from passing through an additive white Gaussian noise channel. This channel is characterized with its noise variance σ_w^2. As a matter of clarification, we assume that synchronization is perfect, that the resulting signal is demodulated and that we access LLR of the intercepted bits. Finally, the intercepted noisy signal $\mathbf{y}$ is made up of N samples:

$$y(k) = s(k) + w(k), \forall k \in [\![0, N-1]\!] \tag{1}$$

where $s(k)$ (resp. $w(k)$) is the k^{th} elements of the sent signal (resp. the noise). The sequence $\mathbf{s}$ results from a concatenation of BPSK-modulated code words: for all k, $s(k)$ is in $\{\pm 1\}$. For instance, the i^{th} vector of n consecutive samples is denoted $\mathbf{s}_i = [s(i \cdot n), \cdots, s(i \cdot n + n - 1)]$. The received signal samples $y(k)$ are proportional to the LLR of the sent samples $s(k)$. Finally, we assume that the synchronization is known. More specifically, the received data stream begin with a full code word.

In the following, we denote by $\mathbb{P}(E)$ and by $\mathbb{E}(X)$, the probability of an event E and the expectation of a random variable X respectively. A random variable is always represented by a capital letter while lower case letter stands for the corresponding deterministic value. The uniform distribution on a set I is identified by $\mathcal{U}(I)$. Likewise, $\mathcal{N}(\mu, \sigma^2)$ stands for the normal distribution of mean μ and variance σ^2 whereas $\chi^2(n)$ describes a chi-squared distribution with n degrees of freedom. Moreover, Euclidean distances between two vectors $\mathbf{a}$ and $\mathbf{b}$ is written as $d_E(\mathbf{a}, \mathbf{b})$.

B. Number of Classes Deficiency

Our method is based on geometric considerations. Intuitively, if we were able to represent an infinite number of noisy received code words in a n_c-dimensional Euclidean space, their positions would be altered by the AWGN channel effects. In fact, each code word would "move to a nearby point according to a spherical Gaussian distribution" [23]. Nevertheless, we would notice the formation of agglomerates (if the noise power is not too high). These agglomerates concentrate around the initial positions of the non-noisy code words. Furthermore, we can predict that this space is sparse compared to the i.i.d. case. Indeed, due to the redundancy induced by the encoder, only 2^{k_c} possible code words are placed in a space which is able to contain 2^{n_c}. To illustrate this phenomena, Figure 2 compares two distribution cases in a 3-dimensional Euclidean space: without encoding 2(a), with encoding 2(b). In the case 2(b) it visually seems to be easier to discriminate the agglomerates (and thus the initial code words positions) thanks to the the sparsity despite the effects of noise.

In this paper, we aim to detect the code length n_c through a classification process. Each agglomerate is a class. A first definition of the *Number of Classes Deficiency* could be: it is a phenomena due to the redundancy induced by the encoder where the quantity of classes is less than expected in a given n-dimensional Euclidean space. We need to quantify this deficiency:

Definition 1. *Number of Classes Deficiency (NCD):*
Let us define random variables $Z^{(n)}$ counting the number of classes in a given n-dimensional Euclidean space for a classification of i.i.d. n-tuples and $z_c(n)$ the number of classes observed from the encoded stream. The deficiency is:

$$\Delta_{NCD}(n) = \mathbb{E}(Z^{(n)}) - z_c(n) \tag{2}$$

In the next section, we develop our criterion based on Definition 1 to identify the code length n_c.

II. The code length estimation: n_c

A. The classification process

To estimate the code length, we propose to find the dimension n for which we have the larger class deficiency. For this purpose, the noisy intercepted stream of N samples is split in L contiguous blocks $B_i^{(n)}$ of length n as described by equality (3), with $L = \lfloor \frac{N}{n} \rfloor$ and $i \in [\![0, L-1]\!]$.

$$B_i^{(n)} = [y(i \cdot n), \cdots, y(i \cdot n + n - 1)] \tag{3}$$

Then, a classification process based on a Euclidean distance criterion is operated on these blocks through two main steps for each value of n:

1. The first block $B_0^{(n)}$ defines the first element of the first class: we name it a *reference word* R_0. For each $i \neq 0$, the following blocks $B_i^{(n)}$ are compared to R_0: if the Euclidean distance between R_0 and $B_i^{(n)}$ is below a chosen threshold β, then they are parts of the same class. Otherwise, if this distance is larger than 2β, then $B_i^{(n)}$ becomes the reference word of an other class. When a distance lies between β and 2β, the involved block is put aside until the others has been handled. This way of choosing the reference words allows to minimize the *class overlapping*.
2. The put aside blocks are classified with the same process. Nevertheless, the choice of a new reference word is less constrained. Its distance to an existing reference word can lie between β and 2β.

Once it enters a class, a block is removed from the set of all the created blocks, i.e. each block belongs to only one class. Finally, for each n, we get access to the number of classes $z_c(n)$ such as:

$$z_c(n) = \mathrm{Card}(\mathcal{R}^{(n)}) \tag{4}$$

where $\mathcal{R}^{(n)}$ is the set of all reference words for a given n, and $\mathrm{Card}(\mathcal{R}^{(n)})$ his cardinal.

A first appreciation of the results given by the classification process is depicted with Figure 3(a). Here, the chosen encoder is an irregular LDPC code which length and rate are $n_c = 25$ and $\rho = \frac{2}{5}$ respectively. For this example, $N = 25000$ bits have been sent. The expectation of the number of classes in the absence of encoding (black squares) and the numbers of classes obtained in the encoded case (red cercles) for each block size n are compared. We notice that the respective amounts of classes in the coded and in the uncoded cases roughly match for each block size n, except for one. At $n = n_c = 25$, the code generates an NCD phenomenon. Due to the limitation on the number of received bits N, for n lying between 13 and 30, the value of $\mathbb{E}(Z^{(n)})$ decreases. More precisely, in this case $\lfloor \frac{N}{n} \rfloor$ is an upper bound of $\mathbb{E}(Z^{(n)})$.

Nevertheless, because of the particular shape of the curve, this result is not directly exploitable. Moreover, we notice

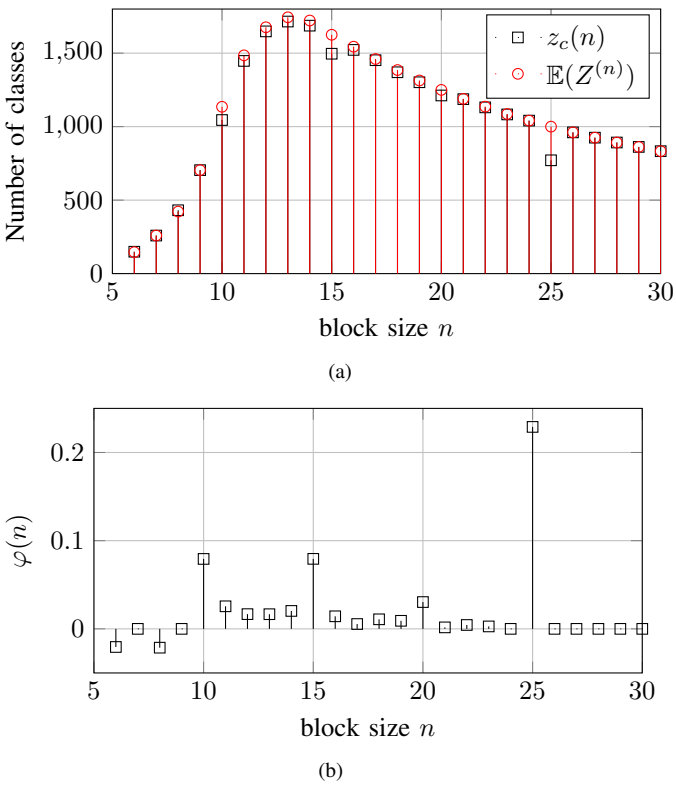


Fig. 3. The NCD highlighting: (a) comparison of the $\mathbb{E}(Z^{(n)})$ and $z_c(n)$, (b) normalized NCD

some subsidiary deficiencies for different values of n between 10 and 15. These are due to the particular structure of the considered LDPC code. To automatically detect the preponderant deficiency, we implement a normalized NCD (Definition 2). The effects of this normalization are illustrated by Figure 3(b) which reveals the most significant deficiency.

Definition 2. *The Normalized Number of Classes Deficiency* φ*:*

$$\begin{aligned} \varphi(n) &= \frac{\mathbb{E}(Z^{(n)}) - z_c(n)}{\mathbb{E}(Z^{(n)})} \\ &= \frac{\Delta_{NCD}(n)}{\mathbb{E}(Z^{(n)})} \end{aligned} \tag{5}$$

This definition leads to the following detection criterion:

$$\hat{n}_c = \arg\max_n(\varphi(n)) \tag{6}$$

In this presentation of the principle of our algorithm, we willingly skipped the definition of the threshold β. It seems obvious that the choice of this parameter value is central in our method. Consequently, in the following, we highlight the dependencies to the threshold β: the number of classes observed from the encoded stream $z_c(n)$ and the random variable $Z^{(n)}$ are denoted $z_c(n, \beta)$ and $Z_\beta^{(n)}$ respectively. The normalized NCD becomes: $\varphi(n, \beta)$.

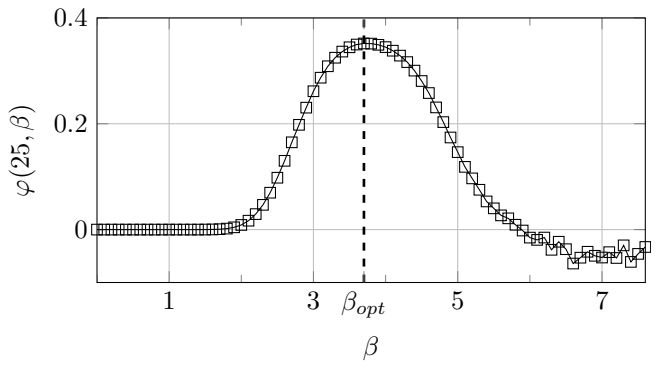


Fig. 4. Normalized deficiency when $n = n_c = 25$ for different values of β

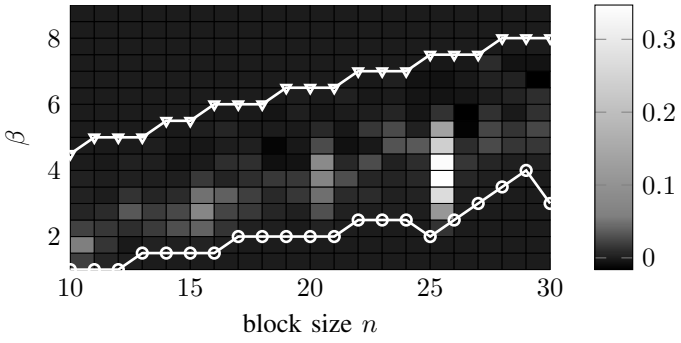


Fig. 5. Observation of $a_\beta(n) \cdot \varphi(n,\beta)$ with the limitation region for the values of β

B. Characterization of the threshold β

At first, the ideal approach should be to directly process the classification algorithm for an optimal threshold β_{opt}. We consider that when the normalized deficiency φ is maximum, β is optimal. For example, with the same code as previously ($n_c = 25$, $k_c = 10$), we can empirically determine the best threshold as showed Figure 4 for $n = n_c$. The optimal threshold is $\beta_{opt} \approx 3.7$. A way to obtain the optimal threshold would be to solve the following problem:

$$\beta_{opt} = \arg\max_{\beta} \mathbb{E}\left(\varphi(n,\beta)\right) \tag{7}$$

Unfortunately, we are not able to provide an analytical expression for β_{opt} since it depends on the code itself. Thereby, instead of choosing a fixed threshold, we sweep on several values with a chosen step.

From Figure 4, we observe that $\varphi(n = n_c, \beta) = 0$ for extreme values of β. On one hand, for the largest thresholds, all blocks are agglomerated in one class in both encoded and non-encoded cases. On the other hand, when β tends to 0, each single block is considered as a class. Hence, we eliminate some values of β to limit the computing effort. First, it seems useless to process the classification for $\beta < \min_{i\neq j}(d_E(B_i^{(n)}, B_j^{(n)}))$. There would be as many classes as created blocks. In this context, it would be impossible to observe an NCD. It provides us the minimum threshold: $\beta_{min}(n) = \min_{i\neq j}(d_E(B_i^{(n)}, B_j^{(n)}))$. Second, there is no use to choose β greater than the average distance between two blocks. Indeed, let us define X_{d^2} the random variable representing the squared Euclidean distance between two randomly chosen blocks. According to an approximation based on the Central Limit Theorem [24], for a large n, X_{d^2} is normally distributed with mean μ_{d^2} and variance $\sigma^2_{d^2}$. Indeed, X_{d^2} is a sum of i.i.d. equally normally distributed random variables (more details are available in Appendix A). In other words:

$$\mathbb{P}(d_E^2(B_i^{(n)}, B_j^{(n)}) \leqslant \mu_{d^2}) = \frac{1}{2} \tag{8}$$

It means that there is one chance over two that two randomly chosen blocks belong to the same class when β^2 reaches μ_{d^2}. For this reason, in our classification process, exceeding $\beta = \sqrt{\mu_{d^2}}$ gives too much agglomeration power to each block. This ascertainment leads us to an upper bound for β: $\beta_{max}(n) = \sqrt{\mu_{d^2}}$. Finally, for a fixed n:

$$\min_{i\neq j}(d_E(B_i^{(n)}, B_j^{(n)})) \leqslant \beta \leqslant \sqrt{\mu_{d^2}} \tag{9}$$

We have now access to several values of $\varphi(n,\beta)$ for β lying between β_{min} and β_{max} for each block size. To take a decision about the preponderance of a deficiency, we define a new normalized NCD:

Definition 3. *The Steady Normalized Number of Classes Deficiency φ_s:*

$$\varphi_s(n) \quad = \sum_{\beta\in[\beta_{min}(n),\,\beta_{max}(n)]} a_\beta(n)\cdot\varphi(n,\beta) \tag{10}$$

with $a_\beta(n)$ the normalization coefficient depending on the value taken by β for a specific n.

$$a_\beta(n) = \frac{Z_\beta^{(n)} - \min_\beta(Z_\beta^{(n)})}{\max_\beta(Z_\beta^{(n)}) - \min_\beta(Z_\beta^{(n)})} \tag{11}$$

Figure 5 represents $a_\beta(n)\cdot\varphi(n,\beta)$ on a grid defined for $n \in [\![10, 30]\!]$ and for β lying between $\beta_{min}(n)$ and $\beta_{max}(n)]$ with a step of $\Delta\beta = 0.5$. More details about the influence of $\Delta\beta$ and its impact on the code length estimation are available in the result section. In Figure 5, the points of greatest amplitude correspond to the block size $n = n_c = 25$ and the threshold lying between $\beta = 2.5$ and $\beta = 5.5$. For $\beta > \beta_{max}$ and $\beta < \beta_{min}$, $a_\beta(n)\cdot\varphi(n,\beta)$ is not estimated (and it is replaced by a 0 in Figure 5). In this example, we observe that the choice of β_{max} and β_{min} reduces the computation time without eliminating any value of interest.

$\varphi_s(n)$ is obtained by summing each column depicted in Figure 5 (see Definition 3). For a fixed n, $\varphi_s(n)$ is the weighted sum of several values of $\varphi(n,\beta)$. Each normalization coefficient $a_\beta(n)$ allows to scale each value of $\varphi(n,\beta)$ according to its statistical reliability (the more created classes, the more reliable the measured deficiency). Finally, an NCD must be both preponderant and reliable to be discriminant. Figure 6 represents φ_s versus n. The greater deficiency is obtained for $n = n_c$. Indeed, the detection criterion derives from the Definition 3:

$$\hat{n}_c = \arg\max_n(\varphi_s(n)) \tag{12}$$

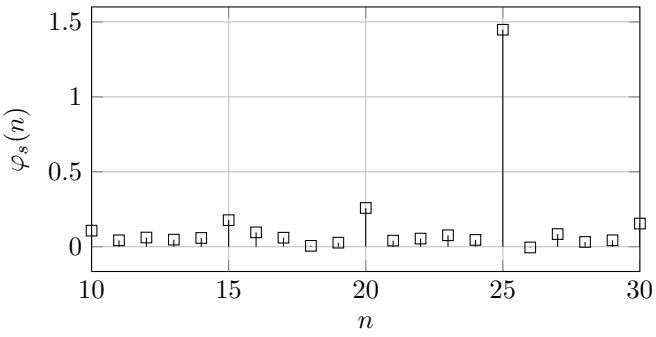


Fig. 6. Steady normalized NCD

C. Blind identification algorithm

The blind identification process is summarized by the Algorithm 1. In this subsection, we give a detailed description of the classification process: the steps 1 and 2. They correspond to the two main steps described in subsection II-A. For a block size n and a distance threshold β, the objective is to compute a number of classes $z_c(n,\beta) = \mathrm{Card}(\mathcal{R}^{(n)})$. As defined previously, $\mathcal{R}^{(n)}$ is the set of all reference words for given n and β. Since R_z is the reference word of the z^{th} created class: $\mathcal{R}^{(n)} = \{R_z | z \in [\![0, z_c(n,\beta)-1]\!]\}$. These reference words are selected among the L created blocks through step 1 and step 2. Moreover, they agglomerate other blocks: it creates classes. A block is considered as classified when it is a reference word or when it is agglomerated by one of them. For a given n we denote by $\mathcal{I}_b^{(n)}(k)$ a set of blocks subscripts. It is defined as follows:

$$\mathcal{I}_b^{(n)}(k) = \{i |\, d_E(B_k^{(n)},\, B_i^{(n)}) \leqslant b\} \tag{13}$$

In step 1, the first reference is the block $R_0 = B_0^{(n)}$ (line 8). $\mathcal{I}_\beta^{(n)}(0)$ represents the subscripts of all the blocks agglomerated by $B_0^{(n)}$. The next reference word R_1 is chosen among the remaining unclassified blocks. Furthermore, to minimize the class overlapping, R_1 must be at a distance of at least 2β from R_0. Finally, every blocks in $\mathcal{I}_{2\beta}^{(n)}(0)$ are ineligible to be the next reference words. In step 1, for each while loop, the classified blocks ($\mathcal{I}_\beta^{(n)}(k)$) and the blocks in $\mathcal{I}_{2\beta}^{(n)}(k)$ are respectively buffered in $\mathcal{F}_\beta^{(n)}$ and in $\mathcal{F}_{2\beta}^{(n)}$ (line 11 and 12 respectively). For each loop, the next chosen reference is the first block distant of at least 2β from every previous references (line 13).

In step 2, the same classification is processed with the remaining unclassified blocks from step 1 (i.e. with the blocks in $\{0,\, 1, \cdots,\, L-1\} \backslash \mathcal{F}_\beta^{(n)}$). The first reference word is chosen line 15 and the following ones are selected line 19. At the end of step 2, every blocks has been classified in $z_c(n,\beta)$ classes.

About the time complexity: The computation cost mainly depends on the amount of operations needed to compute the Euclidean distances. Indeed, for a fixed n, all the two by two distances between blocks are computed. It performs n multiplications and $2n-1$ additions for each of the $\frac{\lfloor N/n \rfloor (\lfloor N/n \rfloor - 1)}{2}$ Euclidean distances. Therefore, about $(3n-1)\frac{\lfloor N/n \rfloor (\lfloor N/n \rfloor - 1)}{2}$ operations are needed to compute the Euclidean distances: this algorithm has a polynomial time complexity of $O(nN^2)$. As a matter of clarification: the amount of intercepted bits has a great influence on the computation time. The more intercepted bits, the more created blocks, the more two by two distances to compute.

Algorithm 1 BLIND IDENTIFICATION ALGORITHM

Require: $\mathbf{y}$, n_{min}, n_{max}

1· **for** $n = n_{min}$ to n_{max} **do**
2· Create L n-length blocks $B_i^{(n)}$ from $\mathbf{y}$ with respect to (3)
3· Compute $\beta_{min}(n)$ and $\beta_{max}(n)$
4· **for** $\beta = \beta_{min}(n)$ to $\beta_{max}(n)$ **do**
5· $\mathcal{F}_{2\beta}^{(n)} = \varnothing$
6· $\mathcal{F}_{\beta}^{(n)} = \varnothing$
7· $\mathcal{R}^{(n)} = \varnothing$
8· $k = 0$
9· **while** $\mathrm{Card}(\mathcal{F}_{2\beta}^{(n)}) < L$ **do**
10· $\mathcal{R}^{(n)} \leftarrow \mathcal{R}^{(n)} \cup \{B_k^{(n)}\}$
11· $\mathcal{F}_{\beta}^{(n)} \leftarrow \mathcal{F}_{\beta}^{(n)} \cup \mathcal{I}_{\beta}^{(n)}(k)$
12· $\mathcal{F}_{2\beta}^{(n)} \leftarrow \mathcal{F}_{2\beta}^{(n)} \cup \mathcal{I}_{2\beta}^{(n)}(k)$
13· $k = \min(\{0,\, 1, \cdots,\, L-1\} \setminus \mathcal{F}_{2\beta}^{(n)})$
14· **end while** } Step 1.
15· $k = \min(\{1, \cdots,\, L\} \setminus \mathcal{F}_{\beta}^{(n)})$
16· **while** $\mathrm{Card}(\mathcal{F}_{\beta}^{(n)}) < L$ **do**
17· $\mathcal{R}^{(n)} \leftarrow \mathcal{R}^{(n)} \cup \{B_k^{(n)}\}$
18· $\mathcal{F}_{\beta}^{(n)} \leftarrow \mathcal{F}_{\beta}^{(n)} \cup \mathcal{I}_{\beta}^{(n)}(k)$
19· $k = \min(\{0,\, 1, \cdots,\, L-1\} \setminus \mathcal{F}_{\beta}^{(n)})$
20· **end while** } Step 2.
21· $z_c(n,\, \beta) = \mathrm{Card}(\mathcal{R}^{(n)})$
22· Compute $\varphi(n,\, \beta)$
23· **end for**
24· Compute $\varphi_s(n)$
25· **end for**
26· $\hat{n}_c = \arg\max_n(\varphi_s(n))$

Ensure: $\hat{n}_c$

In comparison, the method proposed in [2] is based on a Gauss-Jordan elimination. This algorithm is known to have a polynomial complexity of $O(n^3)$, where n is the size of the consecutive created matrices. To work properly our method needs that the amount of created blocks to be large enough: $n \ll L$. As a conclusion the elimination based algorithm is less complex than the deficiency based algorithm since $n^2 \ll N$. However, in the result section we show that this loss in complexity is compensated by an enhanced efficiency.

As an illustration, we propose some tests about the computation time in the result section.

III. THE CODE DIMENSION ESTIMATION: k_c

In this section, we propose a method to estimate the code dimension k_c. From the blind identification algorithm, we suppose that we have now access to the correct code length n_c. With a perfect classification process, we could assume that

each created class is a reliable representation of one given code word. This assumption allows us to see the number of classes as the number of different intercepted code words. Therefore, with infinite observation time, the measured deficiency should be $2^{n_c} - 2^{k_c}$ in a n_c-dimensional space where there is only 2^{k_c} code words and we would immediately have an estimate of k_c.

Since the channel observation time is finite, we would probably not get one representation of each code word. Furthermore, our classification process does not provide a unique value for the deficiency. In this context, we use some results of the well known *birthday problem* by counting *collisions* as in [1]. Indeed, because of the redundancy induced by the encoder, the expected number of collisions is related to k_c. When the distance between two blocks of size n_c is less than a given threshold β_{col}, we consider that there is a collision. In the next subsections, we exhibit the expectation of the number of collisions and we define β_{col}.

A. Expectation of the number of collisions $\mathbb{E}(X_{col})$

In absence of noise, for $K = 2^{k_c}$ possible code words, the probability that a random pair of blocks comes from the same code word is $\frac{1}{K}$. For $L = \lfloor \frac{N}{n_c} \rfloor$ intercepted code words, there is $\frac{L\cdot(L-1)}{2}$ computable distances to detect the collisions. The resulting expectation for the value of the number of collisions is: $\mathbb{E}(X_{col}) = \frac{L\cdot(L-1)}{2}\frac{1}{K}$. In our case, we take noise into account. Let denote by $\mathbf{P}_{col} = \mathbb{P}(d_E(B_i^{(n_c)}, B_j^{(n_c)}) \leq \beta_{col})$ the probability that two blocks $B_i^{(n_c)}$ and $B_j^{(n_c)}$ are in the same class (i.e. collide). Therefore, the expectation for the total number of collisions X_{col} becomes:

$$\mathbb{E}(X_{col}) = \frac{L(L-1)}{2} \cdot \mathbf{P}_{col} \tag{14}$$

From the total probability theorem:

$$\mathbf{P}_{col} = \mathbb{P}(\mathbf{s}_i = \mathbf{s}_j)\mathbf{P}_d + \mathbb{P}(\mathbf{s}_i \neq \mathbf{s}_j)\mathbf{P}_{fa} \tag{15}$$

with:

$$\mathbb{P}(\mathbf{s}_i = \mathbf{s}_j) = 1 - \mathbb{P}(\mathbf{s}_i \neq \mathbf{s}_j) = \frac{1}{K}$$

$\mathbf{P}_d$ represents the probability of true collision, i.e. the probability that two blocks collide given that they both are noisy versions of the same code word. For our purpose, those two blocks are meant to collide. The expression of the probability $\mathbf{P}_d$ derives from a chi-squared distribution with n_c degrees of freedom:

$$\begin{aligned}\mathbf{P}_d &= \mathbb{P}(d_E(B_i^{(n_c)}, B_j^{(n_c)}) \leq \beta_{col} | \, \mathbf{s}_i = \mathbf{s}_j) \\ &= \frac{\gamma\left(\frac{n_c}{2}, \frac{\beta_{col}^2}{4\sigma_w^2}\right)}{\Gamma\left(\frac{n_c}{2}\right)}\end{aligned} \tag{16}$$

Here, $\Gamma(\cdot)$ and $\gamma(\cdot,\cdot)$ respectively denotes the gamma function and the incomplete gamma function (more detail in Appendix B). $\mathbf{P}_{fa}$ is the probability of false collision, i.e. the probability that two blocks collide given that they do not come from the same code word:

$$\mathbf{P}_{fa} = \mathbb{P}(d_E(B_i^{(n_c)}, B_j^{(n_c)}) \leq \beta_{col} | \, \mathbf{s}_i \neq \mathbf{s}_j) \tag{17}$$

TABLE I
VALUES OF THE THRESHOLD β_{col} FOR DIFFERENT E_b/N_0

E_b/N_0	-9	-7	-5	-3	-1	1	3	5
β_{col}	8.5	6.8	5.4	4.3	3.4	2.7	2.1	1.7

Since, we do not know the code words distribution, we can not have an expression for $\mathbf{P}_{fa}$. However, we assume that $\mathbf{P}_{fa}$ is negligible for $\beta = \beta_{col}$, especially for high signal-to-noise ratio (it is discussed in the result section). Therefore, we use the following approximation in our estimation process:

$$\mathbf{P}_{col} \approx \frac{1}{K} \cdot \mathbf{P}_d \tag{18}$$

Finally, from the intercepted sequence, we compute the empirical number of collisions:

$$\hat{x}_{col} = \text{Card}\left(\{(i,j)_{1\leqslant i<j\leqslant L} : \, d_E(B_i^{(n_c)}, B_j^{(n_c)}) \leqslant \beta_{col}\}\right) \tag{19}$$

Therefore, from equations (14) and (18):

$$\hat{x}_{col} \approx \frac{L(L-1)}{2} \cdot \frac{1}{K} \cdot \mathbf{P}_d \tag{20}$$

Since $K = 2^{k_c}$ and k_c is an integer, we propose the following estimator for the code dimension:

$$\hat{k}_c = \left\lfloor \log_2\left(\frac{L(L-1)}{2\hat{x}_{col}} \cdot \mathbf{P}_d\right)\right\rceil \tag{21}$$

where $\lfloor x \rceil$ is the nearest integer to x. The empirical number of collisions $\hat{x}_{col}$ depends on a collision threshold β_{col} which is relative to the probability $\mathbf{P}_d$.

B. Choice of the collision threshold β_{col}

To obtain a representative number of collisions, we aim at finding the distance for which two initially identical code words are colliding. That is, we define this distance as:

$$\beta_{col} \triangleq \underset{\beta\in[\beta_{min},\,\beta_{max}]}{\arg\max} \, f\left(\frac{\beta^2}{2\sigma_w^2}, n_c\right) \tag{22}$$

where $f(x,n)$ is the probability density function associated to the chi-squared distribution with degrees of freedom n. As a matter of clarification:

$$f(x,n) = \frac{\left(\frac{1}{2}\right)^{\frac{n}{2}}}{\Gamma\left(\frac{n}{2}\right)} (x)^{\frac{n}{2}-1} e^{-\frac{x}{2}} \tag{23}$$

The equality (22) defines β_{col} as the distance maximizing the density of truly colliding code words. Even if $\mathbf{P}_{fa}$ is unknown, this value for β_{col} ensure that the number of true collisions is large. Let us now consider the same LDPC code with $n_c = 25$ and $k_c = 10$. For each E_b/N_0 value, the collision threshold β_{col} is defined with respect to (22). Some of these thresholds are gathered in Table I.

The greater the noise, the higher the collision threshold. Indeed, on one hand, when the amount of noise increases, each code word deviates even more from its original position. It results in the increasing of the mean distance between two

identical code words. On the other hand, the probability of false collision increases, which leads to errors of estimation. It will be discussed in the section IV.

IV. Simulation results

In this section, we evaluate the performance of the proposed methods. We highlight the performance of our algorithm by confronting it to a method based on hard decision taken over the received sequence of bits [2]. In the following, GEADC refers to this methods and stands for Gaussian Elimination with Almost Dependent Columns. To illustrate our results, we use three LDPC codes, C_1: ($n_c = 25$, $k_c = 10$), C_2: ($n_c = 25$, $k_c = 20$) and C_3: ($n_c = 15$, $k_c = 10$). Their respective rates are $\frac{2}{5}$, $\frac{4}{5}$ and $\frac{2}{3}$. The choice of this three codes is motivated by the will to apprehend the influence of the code rate on the performance of our method. To test the time complexity of our algorithm, we also give figures for a longer LDPC code C_4: ($n_c = 96$, $k_c = 48$).

A. Detection of n_c

In this subsection, we check that our method is relevant. For C_1, C_2 and C_3, we compare the probability of correct estimation for the code length (denoted $\mathbf{P}_{n_c}$) with our method on one side (denoted NCD), and with the GEADC method on the other side for 1000 code words. The GEADC method is computed for 5 virtualization iterations. At each iteration, the algorithm operates a random permutation on the lines of the interception matrix. This permutation improves the estimation because the Gauss elimination is performed on another set of data. A thousand Monte-Carlo trials have been run to plot the curves in Figure 7: 7(a), 7(b) and 7(c) are respectively concerned with C_1, C_2 and C_3. During each trial, a new intercepted bits stream has been randomly generated: noise and information bits. The grid step for β is fixed at 0.5.

From Figure 7, we observe that our method based on LLRs outperforms the Gauss reduction method whatever the code chosen. For a probability of correct estimation close to 0.8, the gain with our method is about 4 dB for C_1, and 3 dB for C_2 and C_3. Furthermore, we notice that the smaller the code rate, the better the performance. On one side, when n_c is fixed ($n_c = 25$ for C_1 and C_2), the mean distance between classes tends to be greater for the code with fewer code words (C_1). On the other side, when k_c is fixed ($k_c = 10$ for C_1 and C_3), the mean distance is greater when the code length grows since all the code words are on the surface of a n_c-sphere (with radius $\sqrt{n_c}$).

1) About the observation time: The observation time has a significant impact on the probability of detection. Indeed, the more data, the more accurate the detection. For different signal-to-noise ratio, Figure 8 shows the detection probability versus the number of intercepted code words $\lfloor \frac{N}{n_c} \rfloor$ for C_1, C_2 and C_3. For each code, we choose two points of interest. For 1000 code words and $\Delta\beta = 0.5$, we focus on the lowest E_b/N_0 for which the algorithm reaches $\mathbf{P}_{n_c} = 1$ and the one for $\mathbf{P}_{n_c} \approx 0.5$. It allows us to observe gains (or losses) when increasing (or decreasing respectively) the length of intercepted sequence. All the three codes have the same

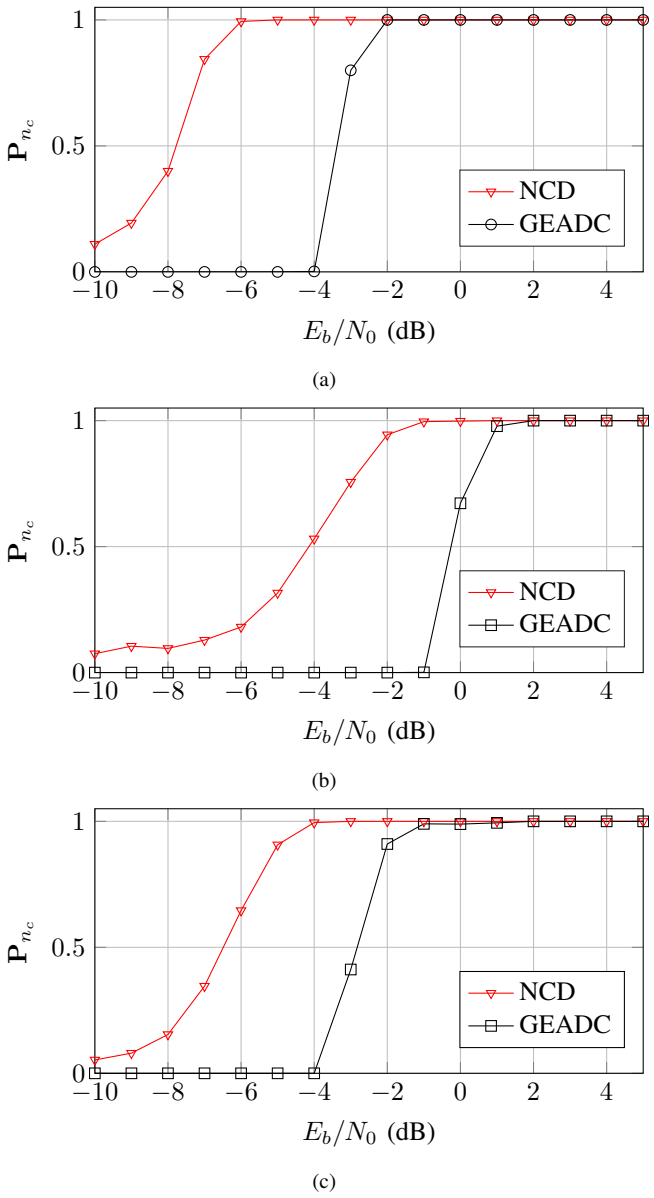


Fig. 7. Comparison of the soft and hard information based methods (red triangles and black circles respectively) for C_1 (7(a)), C_2 (7(b)) and C_3 (7(c))

behavior. For instance, doubling the amount $\lfloor \frac{N}{n_c} \rfloor$ from 1000 to 2000 almost double the probability of correct identification for C_1 and C_3 at lowest E_b/N_0s. Both codes have the same dimension $k_c = 10$ and these orders of magnitude for $\lfloor \frac{N}{n_c} \rfloor$ deeply affects the population inside classes since there are 2^{10} different code words. For C_2 ($k_c = 20$), the effect is less significant: more words are needed to reach the same performance. Reducing the quantity of received code words has the opposite impact: the population inside classes decreases and the distribution of the blocks in the n-dimensional space tends to be uniform even if $n = n_c$.

Finally, maximizing the number of treated bits improves the performance of our algorithm. Nevertheless, the computation time increases significantly with the quantity of intercepted bits. Indeed, at each iteration on the block size n, the number

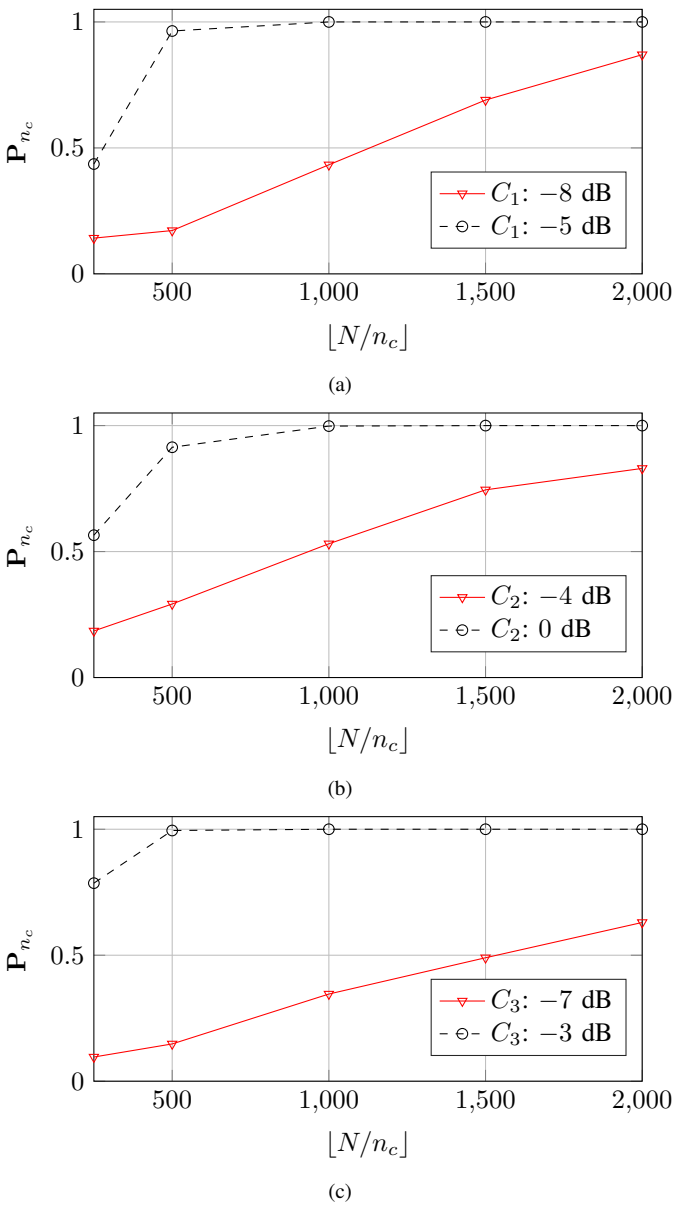


Fig. 8. Impact of the quantity of received code words for C_1 (8(a)), C_2 (8(b)) and C_3 (8(c))

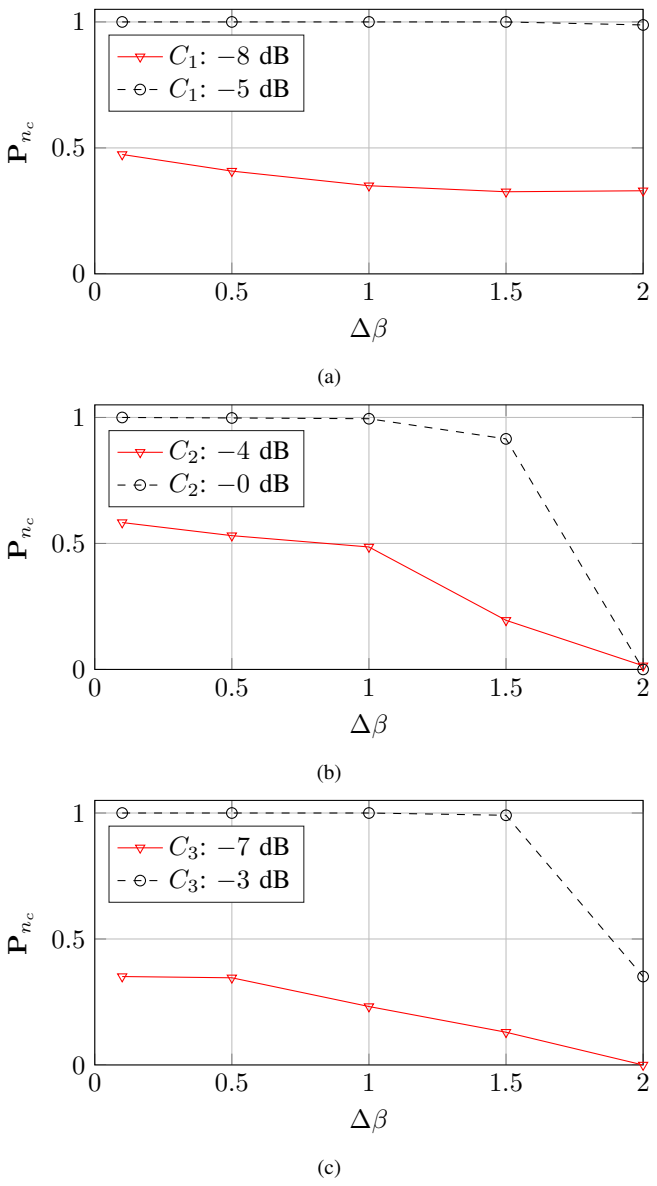


Fig. 9. Influence of the threshold step on the detection for C_1 (9(a)), C_2 (9(b)) and C_3 (9(c))

of computed distances is $\frac{\lfloor \frac{N}{n} \rfloor (\lfloor \frac{N}{n} \rfloor - 1)}{2}$.

2) About the threshold grid step size $\Delta\beta$: The quality of the code length estimation also depends on the choice of the threshold β. However, being unable to theoretically determine an optimal threshold, we explore and compute $\varphi(n, \beta)$ for several values of β. It leads us to the choice of a grid step $\Delta\beta$. Figure 9 depicts the impact of $\Delta\beta$ on the probability of detection for C_1, C_2 and C_3 at the same respective signal-to-noise ratios as previously. The thinner $\Delta\beta$, the better the detection, but it asks then a bigger computing effort. The choice of the threshold step is related to the code rate. For a fixed n_c, when the code rate gets larger, the redundancy decreases. In this case, it is not possible to roughly look for classes because there are fewer unoccupied locations in the n_c-dimensional space. Here, C_2 requires to choose the thinnest $\Delta\beta$ to reach the optimal performance of our algorithm while $\Delta\beta$ has no significant impact in the case of C_1. On the other side, it is possible to reduce the complexity for C_1 by choosing a larger $\Delta\beta$ without reducing the efficiency. The optimal threshold step should ensure a correct identification while minimizing the computing effort. As a consequence, for increasing rate, decreasing threshold step are needed. However, according to Figure 9, when $\Delta\beta < 1$, it has little influence on $\mathbf{P}_{n_c}$.

3) Performance for unsynchronized bits streams: The algorithm presented here, is sensitive to the frame synchronization. Since it relies on the computation of Euclidean distances, two blocks are likely to be separated in different classes even if they share identical parts of the same code word. Figure 10 shows the impact of an unsynchronized sequence on the detection performance for the three codes (C_1, C_2 and C_3). For

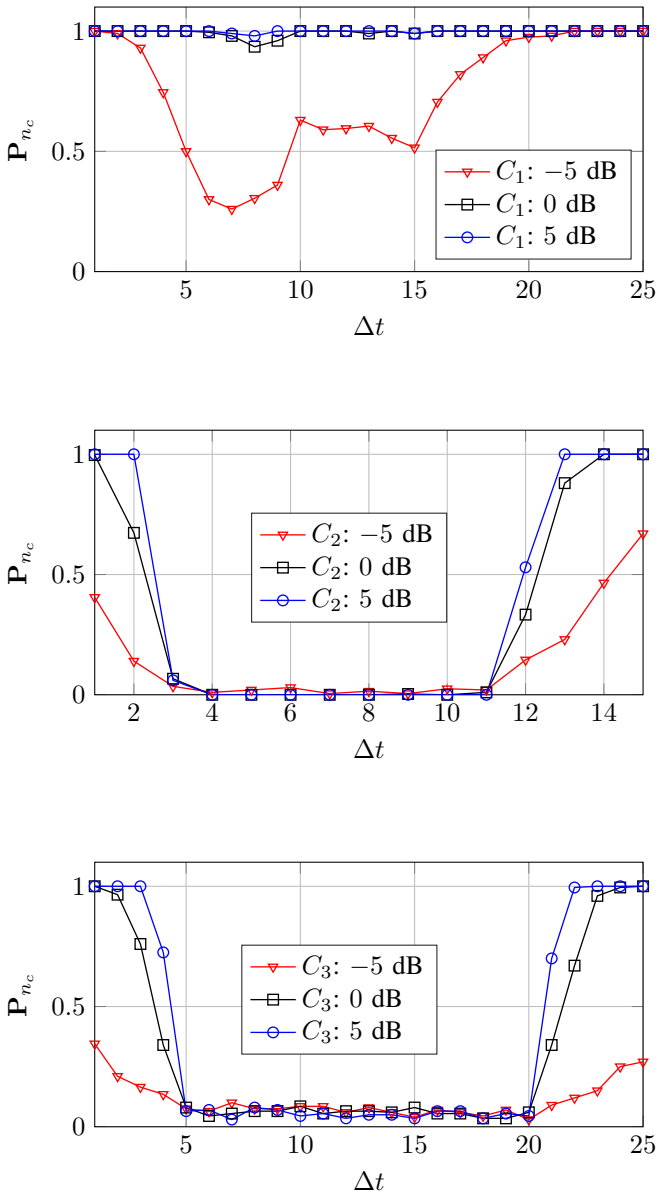


Fig. 10. Impact of the frame synchronization on the code length estimation

C_2 and C_3, as expected, our method performs well around the synchronization region (when Δt tends to n_c or to 0). For C_1, the method performs better whatever the synchronization Δt. Indeed, C_1 is a quasi-cyclic code with parameter 5. Finally, the robustness of the method to the frame synchronization depends on the type of the code. To find the beginning of a code word, a solution would be to run the algorithm for different values of Δt.

4) About the computation time and complexity: We ran a simulation for a longer code (C_4: $n_c = 96$, $k_c = 48$) to show that our method can deal with codes having a length of hundreds of bits. The main issue is the size of the intercepted sequence of bits. Simulation has been run with a Intel(R) Xeon(R) CPU E5-2670 2.60GHz configuration and Matlab. Neither optimization of the Matlab code nor parallelization have been done. It is just an illustration of the evolution

TABLE II
PROBABILITY OF CORRECT IDENTIFICATION AND COMPUTATION TIME FOR A LDPC CODE $(96, 48)$

$\lfloor N/n_c \rfloor$	9000	11000	15000	20000	24000
T (min)	9	15	28	60	100
$\mathbf{P}_{n_c}$	0.55	0.69	0.89	0.99	1

of the computation time versus the number of intercepted bits. Despite the length of the code, it is still possible to detect it perfectly when the amount of received code words is $L = 24000$, i.e. 2304000 bits. Table II synthesizes the results in terms of computation time T in minutes and probability of correct estimation $\mathbf{P}_{n_c}$. At a E_b/N_0 of 10 dB and for 100 Monte Carlo trials, we test C_4 from 9000 to 24000 received code words. T is the time for one trial. The major issue is related to the computational complexity: 100 minutes are needed to detect the channel code with 24000 code words. However, this complexity issue is not intractable. It is conceivable to parallelize some processes: for example, the classifications operated for each block size are independent. Identically, within each classification, each threshold β can be tested separately.

B. Estimation of k_c

From the classification process, we have now an estimate of n_c. All the results exposed here rely on the assumption of a perfect estimation of the code length. For 2000 code words, Figure 11 allows us to observe the impact of the noise on the probability of correct estimation of k_c: $\mathbf{P}_{k_c}$. From 1000 Monte-Carlo trials, we notice different behaviors related to the code used. The most accurate estimation results from testing C_1: it has the lowest code rate. With the same dimension, the algorithm performs less with C_3 because of its larger code rate. In the case of C_2, the amount of received code words is too low when $\lfloor \frac{N}{n_c} \rfloor = 2000$. At high E_b/N_0, there is not enough collisions or even no collision at all to estimate k_c. Furthermore, due to its high rate, a slight increase of noise makes different code words collide. Indeed, the $\mathbf{P}_{fa}$ has a greater influence on the $\mathbf{P}_{col}$ according to (15) since $K = 2^{20}$. In the following, we give more details about the observation time and about the impact of the neglected $\mathbf{P}_{fa}$.

1) About the observation time: The dimension estimator efficiency substantially depends on the amount of received code words. As shown in Figure 12, when the quantity of received code words increases the probability of having at least one collision converges to 1. We can also see that the noise has an impact on the required amount of code words needed to properly estimate the code dimension. Indeed, adding noise increases the $\mathbf{P}_{fa}$. As a result, we need more words to retrieve k_c efficiently for C_3 at 4 dB. At 5 dB, 1000 code words are enough to perfectly estimate k_c for both C_1 and C_3 while 1500 code words are needed to reach the same performance for C_3 at 4 dB. This difference of behavior is due to the respective code rate. Indeed, for two codes having the same dimension k_c (e.g. C_1 and C_3), it is easier to estimate k_c for code having the lower rate (here, C_1) since the mean distance increases

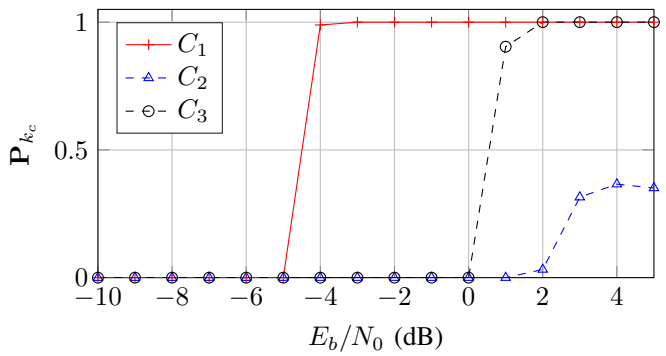


Fig. 11. Probability of correct estimation of the code dimension with noise for C_1, C_2 and C_3 with $\lfloor \frac{N}{n_c} \rfloor$ code words

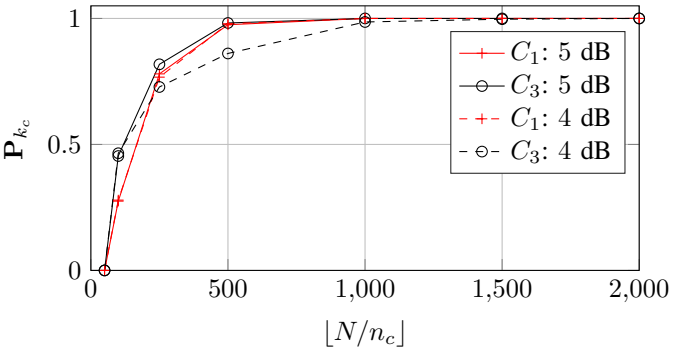


Fig. 12. Probability of correct estimation of k_c for different amounts of received code words for C_1 and C_3 at 5 and 4 dB

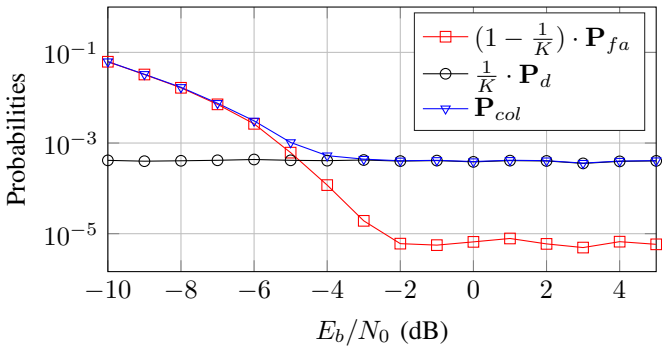


Fig. 13. Comparison of the respective influences of $\mathbf{P}_{fa}$ and $\mathbf{P}_d$ on the probability of collision for different E_b/N_0 values with $\lfloor \frac{N}{n_c} \rfloor = 2000$ for C_1

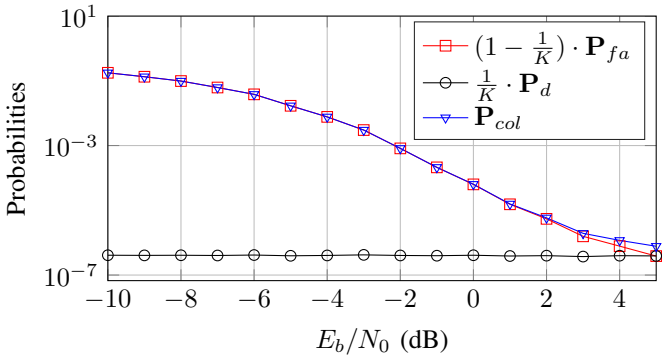


Fig. 14. Comparison of the respective influences of $\mathbf{P}_{fa}$ and $\mathbf{P}_d$ on the probability of collision for different E_b/N_0 values with $\lfloor \frac{N}{n_c} \rfloor = 2000$ for C_2

when the code rate decreases. Notice that performance for C_2 is not given since we would have to intercept much more than 2000 code words in order to estimate k_c for that code (i.e. due to its length and rate).

2) About the impact of the $\mathbf{P}_{fa}$*:* In subsections III-A and III-B, we proposed an approximation for $\mathbf{P}_{col}$ by neglecting $\mathbf{P}_{fa}$ for high E_b/N_0 values. Indeed, the amount of false collisions converges to 0 with decreasing noise. To illustrate that, Figure 13 shows how false and true collisions respectively affects the total probability of collision for different values. For this purpose, we compare $(1-\frac{1}{K})\cdot\mathbf{P}_{fa}$ and $\frac{1}{K}\cdot\mathbf{P}_d$ for C_1 with $L = \lfloor \frac{N}{n_c} \rfloor = 2000$ intercepted code words. Since we have a theoretical expression for $\mathbf{P}_d$ and an estimation of the probability of collision $\hat{\mathbf{P}}_{col} = \frac{2\hat{x}_{col}}{L(L-1)}$, we easily access an estimated value of $\mathbf{P}_{fa}$ from (15). For higher E_b/N_0 values, the impact of the probability of false collision is comparatively negligible in the calculation of $\mathbf{P}_{col}$ with respect to the probability of true collision. We also notice that when $(1-\frac{1}{K})\cdot\mathbf{P}_{fa}$ reaches the order of magnitude of $\frac{1}{K}\cdot\mathbf{P}_d$ at -5 dB, $\mathbf{P}_{k_c}$ drops to 0 at the same E_b/N_0 value as shown on Figure 11. The proportion of false collisions is too high and prevent from estimating k_c.

Now, we perform the same analysis, but for C_2. The estimator is less efficient for C_2 with 2000 code words. On one hand, we see from Figure 11 that $\mathbf{P}_{k_c}$ is about 0.35 at 5 dB. On the other hand, we notice on Figure 14 that $\frac{1}{K}\cdot\mathbf{P}_d$ is lower than $(1-\frac{1}{K})\cdot\mathbf{P}_{fa}$ for most of the E_b/N_0 value. Moreover, for any E_b/N_0 value, $\mathbf{P}_{col} \approx (1-\frac{1}{K})\cdot\mathbf{P}_{fa}$. In this circumstances, most of the occurring collisions between 2 and 5 dB are false collisions. This behavior is due to the high code rate. Indeed, the higher the code rate, the lower the minimal distance. Any alterations due to the noise implies false collisions with high probability.

TABLE III
PROBABILITY OF CORRECT IDENTIFICATION FOR k_c FOR AN ERRONEOUS ESTIMATED VALUE FOR NOISE VARIANCE

ϵ_{σ_w}	0 %	±1 %	±2 %	±5 %	±7.5%	±10%
$\mathbf{P}_{k_c}$	1	1	1	0.7	0.14	0

3) About the noise variance estimation: The noise variance has an impact on the choice of the threshold β_{col} for the dimension estimation. Therefore, an erroneous estimated value $\hat{\sigma}_w^2$ for the noise variance might affect the estimation of the code dimension. We ran some tests with C_1: we artificially generated errors $\epsilon_{\hat{\sigma}_w^2} = \pm 1\%, \pm 2\%, \pm 5\%, \pm 10\%$ (e.g. $\epsilon_{\hat{\sigma}_w^2} = \pm 1\%$ means that $\hat{\sigma}_w^2 = 0.99\sigma_w^2$ or $\hat{\sigma}_w^2 = 1.01\sigma_w^2$). Table III sums up the results by giving the probability of correct identification $\mathbf{P}_{k_c}$ and the corresponding threshold for a E_b/N_0 of 0 dB:

We notice that minor errors in the estimation of the noise

variance does not alter the performance.

V. Conclusion

In this paper, we proposed the design of a new algorithm based on soft information. The method presented here allows to blindly recover the length and the dimension of a code from a noisy intercepted bit stream. The identification of the code length relies on a classification method using Euclidean distances as criterion. This method highlights a difference of behavior in the number of classes created between a coded data stream and an i.i.d. sequence when increasing the size of the classes. In presence of a code, the theoretical number of classes is limited by the length of the information words. To retrieve this length, we also adapt a result from the birthday problem and proposed an estimator. The expected number of collision depends on the code dimension. Both length and dimension estimation rely on the observation time and on the amount of noise.

The method described is applied to linear block codes. No a priori knowledge about their construction or their family is needed. By considering soft-decision instead of hard-decision, we aimed to reproduce the performance gap between soft and hard decoding. From our result, we showed that our method performs better than the Gaussian elimination based on almost dependent columns in [2]. Several improvements are to be investigated. For now, the classification process is quite naive: by choosing more reliable reference words instead of random ones, it might be more efficient. Also, due to its complexity, this method is particularly adapted to low length codes: it could be interesting to apply our procedure to convolutional codes.

Appendix A
Details on the Central Limit Theorem Approximation

Let us define $Y(k) = S(k) + W(k)$ the random variable carrying the k^{th} bit likelihood value of the received sequence. $S(k) \sim \mathcal{U}(\{-1, +1\}$ and $W(k) \sim \mathcal{N}(0, \sigma_w^2)$. If we consider $X(p) = (Y(i \cdot n + p) - Y(j \cdot n + p))^2$, then $X(0), X(1), X(2), \cdots, X(n-1)$ is a sequence of i.i.d. random variables. These variables are all mutually independent and each of them has the same probability distribution as the others. Let us denote by μ_X and by σ_X^2 the expected value and variance of this distribution. From an approximation based on the Central Limit Theorem [24], for a large n, it is possible to consider $A_n = X(0) + X(1) + \cdots + X(n-1)$ as a normally distributed variable of expected value $n\mu_X$ and variance $n\sigma_X^2$. Consequently, $X_{d^2} = \sum_{p=0}^{n-1}(Y(i \cdot n + p) - Y(j \cdot n + p))^2$ is normally distributed with mean $\mu_{d^2} = n \cdot \mu_X$ and variance $\sigma_{d^2}^2 = n \cdot \sigma_X^2$.

Appendix B
Details about the probability of true collision $\mathbf{P}_d$

Here, the same notations as in Appendix A are used.

$$\begin{aligned}\mathbf{P}_d &= \mathbb{P}(d_E(B_i^{(n_c)}, B_j^{(n_c)}) \leq \beta_{col} | \, \mathbf{s}_i = \mathbf{s}_j) \\ &= \mathbb{P}(X_{d^2} \leq \beta_{col}^2 | \, \mathbf{s}_i = \mathbf{s}_j) \\ &= \mathbb{P}(\sum_{p=0}^{n_c-1} X(p) | \, \mathbf{s}_i = \mathbf{s}_j)\end{aligned}$$

Considering the condition $\mathbf{s}_i = \mathbf{s}_j$, $X(p) = (W(i \cdot n + p) - W(j \cdot n + p))^2$. In addition, $W(i \cdot n + p) - W(j \cdot n + p)$ is normally distributed with mean $\mu_X = 0$ and variance $\sigma_X^2 = 2\sigma_w^2$. In this context, we conclude that $\frac{X_{d^2}}{2\sigma_w^2} \sim \chi^2(n_c)$. Hence, the cumulative distribution function:

$$\mathbf{P}_d = \frac{\gamma\left(\frac{n_c}{2}, \frac{\beta_{col}^2}{4\sigma_w^2}\right)}{\Gamma\left(\frac{n_c}{2}\right)}$$

where $\Gamma(\cdot)$ and $\gamma(\cdot, \cdot)$, respectively stand for the gamma function:

$$\Gamma(x) = \int_0^{+\infty} t^{x-1} e^{-t} dt$$

and the incomplete gamma function:

$$\gamma(a, x) = \int_0^{x} t^{a-1} e^{-t} dt$$

References

[1] M. Bellard and J.-P. Tillich, "Detecting and reconstructing an unknown convolutional code by counting collisions." ISIT, 2014, pp. 2967–2971.

[2] G. Sicot, S. Houcke, and J. Barbier, "Blind Detection of interleaver parameters," *ELSEVIER Signal Processing*, vol. 20, pp. 450–462, Nov. 2008.

[3] K. Carrier and J.-P. Tillich, "Identifying an unknown code by partial Gaussian elimination." WCC, 2017.

[4] A. Valembois, "Detection and recognition of a binary linear code," *ELSEVIER Descrete Applied Mathematics 111*, pp. 199–218, 2001.

[5] M. Cluzeau and J.-P. Tillich, "On the Code Reverse Engineering Problem." ISIT, 2008.

[6] M. Cluzeau and M. Finiasz, "Recovering a Code's Length and Synchronization from a Noisy Intercepted Bitstream." ISIT, 2009.

[7] J. Barbier and J. Letessier, "Forward Error Correcting Codes Characterization Based on Rank Properties." International Conference on Wireless Communications & Signal Processing, 2009.

[8] J. Barbier and G. Sicot and S. Houcke, "Algebraich Approach for the Reconstruction of Linear and Convolutional Error Correcting Codes," vol. 16. Proceedings of World Academy of Science, Engineering and Technology, 2006.

[9] R. Swaminathan and A. S. Madhukumar, "Classification of error correcting codes and estimation of interleaver parameters in a noisy transmission environment," *IEEE Transactions on Broadcasting*, vol. 63, no. 3, pp. 463–478, Sept 2017.

[10] A. Tixier, "Blind identification of an unknown interleaved convolutional code." China, Hong-Kong: ISIT, 2015, pp. 71–75.

[11] M. Marazin and R. Gautier and G. Burel, "Blind recovery of k/n rate convolutional encoders in a noisy environment." EURASIP Journal on Wireless Communications and Networking, 2011.

[12] M. Côte and N. Sendrier, "Reconstruction of convolutional codes from noisy observation." ISIT, 2009, pp. 546–550.

[13] F. Wang and Z. Huang and Y. Zhou, "A Method for Blind Recognition of Convolution Code Based on Euclidean Algorithm." WiCom, 2007, pp. 1414–1417.

[14] J. Dingel and J. Hagenauer, "Parameter Estimation of a Convolutional Encoder from Noisy Observation." ISIT, June 2007, pp. 1776–1780.

[15] P. Lu and L. Shen and X. Luo and Y. Zou, "Blind Recognition of Punctured Convolutional Codes." ISIT, July 2004.

[16] M. Côte and N. Sendrier, "Reconstruction of a turbo-code interleaver from noisy observation." ISIT, June 2010, pp. 2003–2007.

[17] Y.G. Debessu and H.C. Wu and H. Jiang, "Novel Blind Encoder Parameter Estimation for Turbo Codes," *Communication Letters*, vol. 16, no. 12, pp. 1917–1920, Dec. 2012.

[18] H. Zhang and H.-C. Wu and H. Jiang, "Novel Blind Encoder Identification of Reed-Solomon Codes with Low Computational Complexity." GLOBECOM, 2013.
[19] A. D. Yardi, S. Vijayakumaran, and A. Kumar, "Blind reconstruction of binary cyclic codes from unsynchronized bitstream," *IEEE Transactions on Communications*, vol. 64, no. 7, pp. 2693–2706, July 2016.
[20] R. Moosavi and E. G. Larsson, "Fast Blind Recognition of Channel Codes," *IEEE Transactions on Communications*, vol. 62, no. 5, pp. 1393–1305, 2014.
[21] T. Xia and H.-C. Wu, "Novel Blind Identification of LDPC Codes Using Average LLR of Syndrome *a Posteriori Probability*," *IEEE Transaction on Signal Processing*, vol. 62, no. 3, pp. 632–640, Feb. 2014.
[22] P. Yu, J. Li, and H. Peng, "A least square method for parameter estimation of rsc sub-codes of turbo codes," *IEEE Communications Letters*, vol. 18, no. 4, pp. 644–647, April 2014.
[23] C. E. Shannon, "Probability of Error for Optimal Codes in a Gaussian Channel," *The Bell System Technical Journal*, vol. 38, no. 3, pp. 611–656, 1959.
[24] D. P. Bertsekas and J.N. Tsitsiklis, *Introduction to Probability, SECOND EDITION*, 2008.

Aurélien Bonvard received the M.Sc. degree in Embedded Electronics and Communication Systems from the University of Paris Ouest Nanterre La Défense, France in 2014. His activities were concerned with automatic control systems around the powertrain for the automotive industry. He is currently persuing a Ph.D. degree in Signal Processing at the IMT Atlantique in the Signal and Communications department. His research interests focus on the blind identification and reconstruction of channel codes.

Sébastien Houcke received the Biomedical Ing. degree in biomedical engineering from the University of Technology of Compiegne, Compiegne, France. In 1999, he received the MSc degree in image and signal processing from the University of Cergy-Pointoise, Cergy-Pointoise, France. In 2002, he received his Ph.D. degree at the University of Marne-la-Vallée, Marne-la-Vallée, France. He is currently associate professor at the IMT Atlantique in the Signal and Communications department. His research interests are in signal processing for digital communications with emphasis on blind approaches.

Roland Gautier received the M.Sc degree from the University of Nice-Sophia Antipolis, France, in 1995, where his research activities were concerned with the blind source separation of convolutive mixtures for MIMO systems in digital communications. He received the Ph.D degree in electrical engineering from the University of Nice-Sophia Antipolis, France, in 2000, where his research interests were in experiment design for nonlinear parameters models. From 2000 to 2001, he was Assistant Professor with Polytech'Nantes, the engineering school of the University of Nantes, France. Since September 2001, he has worked with the University of Brest, France, as an Assistant Professor of electronic engineering. His general interests lie in the area of signal processing and digital communications. His current research focuses on digital communication interception, analysis, and blind parameters recognition, Multiple-Access and Spread Spectrum transmissions, Cognitive and Software Radio. From 2007 to June 2012, he was assistant manager of the Signal Processing Group, within the Laboratory for Science and Technologies of Information, Communication and Knowledge (Lab-STICC - UMR CNRS 6285). Since July 2012, he has been the manager of the Intelligence and Furtiveness of Communications Group, within the Lab-STICC. He received his Habilitation to Supervise Research (HDR) from the University of Brest in 2013 presenting an overview of his post-doctoral scientific research activities on the development of self-configuring multi-standard adaptive receivers : Blind analysis of digital transmissions for military communications and Cognitive Radio.

Mélanie Marazin received the M.Sc. degree in sciences and technologies of telecommunications in 2006 from University of Brest, France. Her research activities were concerned with the study of performances of a space-time coding on MIMO channel. She received the Ph.D degree in digital communications from the University of Brest, France, in 2009. Her research interests focus on communications intelligence and blind recognition of convolutional codes. Since 2009, she has Assistant Professor with University of Brest. Her general interests lie in the area of signal processing and digital communications. Her current research focuses on digital communication interception, analysis, and blind parameters recognition.