EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

Blind recognition of binary cyclic codes

Zhou Jing, Huang Zhiping*, Su Shaojing and Yang Shaowu

Abstract

A solution to blind recognition of binary cyclic codes is proposed in this paper. This problem could be addressed on the context of non-cooperative communications or adaptive coding and modulations. We consider it as a reverse engineering problem of error-correcting coding. The proposed algorithm recovers the encoder parameters of a cyclic, coded communication system with the only knowledge of the noisy information streams. By taking advantages of soft-decision outputs of the channel and by employing statistical signal-processing methods, it achieves higher recognition performances than existing algorithms which are based on algebraic approaches in hard-decision situations. By comprehensive simulations, we show that the probability of false estimation of coding parameters of our proposed algorithm is much lower than the existing algorithms, and falls rapidly when signal-to-noise ratio increases.

Keywords: Blind recognition; Channel coding; Cyclic codes; Reverse engineering

1. Introduction

The blind recognition of cyclic codes is a reverse engineering problem of the error-correcting coding which can be applied to non-cooperative communications [1,2] and adaptive coding and modulations (ACM) [3-6]. In most cases of digital communications, forward error-correcting coding is used to protect the transmitted information against noisy channels to reduce errors which occur during transmission. Cyclic codes are one class of the most important error-correcting codes applied in communication area. In cooperative context, the parameters of the codes and modulations are usually known by the transmitters and receivers both. But a receiver in non-cooperative communications or a cognitive radio receiver may not know those parameters and thus cannot directly receive and decode the transmitted information on the channel. Therefore, to adapt itself to an unknown transmission context, the receiver must recognize the modulation and coding parameters blindly before processing the received data. In this paper, we develop an approach for blind recognition of the coding parameters of a communication system which uses binary cyclic codes.

2. Related work

In [7], a Euclidean algorithm-based method is proposed to identify a 1/2-rate convolutional encoder in noiseless cases. However, it is not suitable for noisy channels. In [8], another approach is presented to identify a $1/n$-rate convolutional encoder in noisy cases based on the Expectation Maximization algorithm. The authors of [9,10] develop methods for blind recovery of convolutional encoder in turbo code configuration. In [6,11], a dual code method for blind identification of k/n-rate convolutional codes is proposed for cognitive radio receivers. An iterative decoding-technique-based reconstruction of block code is introduced by the authors of [12] and was applied to low-density parity-check (LDPC) codes. An algebraic approach for the reconstruction of linear and convolutional codes is presented in [13]. In [14], an algorithm for blind recognition of error-correcting codes is presented by utilizing the rank properties of the received stream.

In [15], an approach for blind recognition of binary linear block codes in low code-rate situations is presented. The authors propose to estimate the code length according to the code weight distribution characters of the low-rate codes and then get the generator matrix by improving the traditional simplification of matrices. It has a good performance in high bit error rate (BER) but is not suitable for high code rate situations. Furthermore, it requires a large amount of observed data. In [16] and [17],

* Correspondence: h.zhiping@hotmail.com
Mechatronics Engineering and Automation Department, National University of Defense Technology, Changsha 410073, Hunan Province, People's Republic of China

the authors present a blind recognition algorithm for Bose-Chaudhuri-Hocquenghem (BCH) codes based on the Roots Information Dispersion Entropy and Roots Statistic (RIDERS). This algorithm can achieve correct recognition in both high and low code rate situations with the BER of 10^{-2}. But it is computationally intensive, especially when the code length is large. The authors of [18] improve the algorithm proposed in [16,17] by reducing the computational complexity and making the recognition procedure faster.

Most of the previous works are concentrating on hard-decision situations, and are based on utilizing the algebraic properties of the codes in Galois fields (GF). The major drawback of them is that they have a low fault tolerance. Even if only 1 bit error occurs in a codeword, the algebraic properties of error-correcting codes will be largely destroyed. Therefore, the recognizers need a large amount of observed data. On the other hand, if soft information about the channel output is available, the soft-decision outputs can provide more information for the code recognition, and statistical signal processing algorithms can also be employed to improve the recognition performance.

When statistic and artificial-intelligence-based iterative algorithms are applied to error-correcting decoding, the decoding performance is improved about 2 ~ 3 dB in soft-decision situations [19]. In [20,21], the authors introduce a MAP approach to achieve blind frame synchronization of error-correcting codes with a sparse parity-check matrix. It is also developed on Reed Solomon (RS) codes [22] and BCH product codes [23] and yields better performances than previous hard decision ones. In this paper, we propose an algorithm to achieve blind recognition of binary cyclic codes in soft-decision situations. Literature [4] also considers the blind recognition of coding parameters based on soft decisions. But in fact, its recognition procedure is semi-blind. The authors assume that the channel code which is used at the transmitter is unknown to the receiver, but the code is chosen from a set of possible codes which the authors call *the candidate set*. This set has a limited number of candidates, and is arranged beforehand by both the transmitter and the receiver. It has good performances on ACM, but is not suitable for non-cooperative cases.

To the best of our knowledge, this paper is the first publication to consider the complete-blind recognition problem of binary cyclic codes in soft-decision situations. The proposed algorithm in this paper is based on the RIDERS algorithm introduced in [16-18]. We improve and extend this work in order to handle soft-decision situations. To utilize the soft-decision outputs, we employ the idea of MAP-based processing method proposed in [20-23].

The remainder of this paper is organized as follows: section 3 briefly introduces the RIDERS algorithm in hard decision situations proposed in [16-18]; section 4 presents the principle of our proposed recognition algorithm for binary cyclic codes in soft-decision situation; section 5 draws the general recognition procedure of the proposed algorithm; and finally, the simulation results and conclusions are given in sections 6 and 7.

3. RIDERS algorithm for blind recognition of BCH codes

3.1 Introduction of RIDERS algorithm

The RIDERS algorithm is introduced in [16,17] and improved in [18] to solve the problem of recognition of BCH codes. The system model of blind recognition problem of coding parameters is shown in Figure 1. On the transmitter, the information sequence T_m is encoded and separated to coded blocks T_c by the encoder and modulated before transmitted to the channel. After demodulation, the receiver blindly recognizes the coding parameters and decodes the received blocks R_c to correct the errors which occur during the transmission. R_m is the decoded information which could be processed forward.

We define $c(x)$ to be the codeword polynomial of T_c, then the algebraic model of the encoding procedure can be described as follows [24]:

$$c(x) = m(x) \times g(x) \tag{1}$$

or in systemic form:

$$c(x) = m(x) \times x^{n-k} + \left(\left(m(x) \times x^{n-k} \right) \bmod g(x) \right). \tag{2}$$

where $m(x)$ is the input information polynomial and $g(x)$ is the generator polynomial. The purpose of the

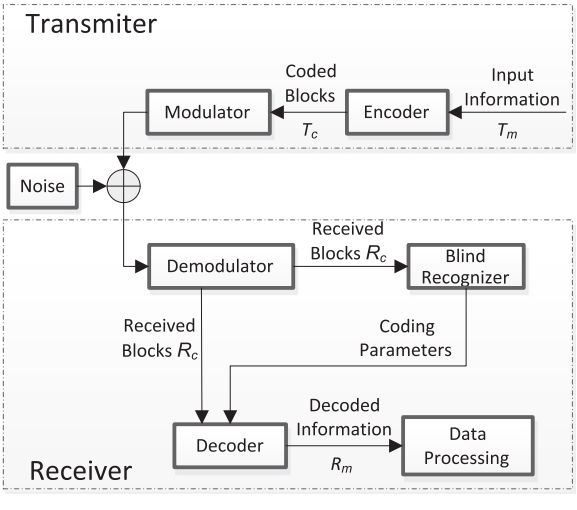

Figure 1 The system model of the blind recognition problem of coding parameters.

recognition is to estimate the codeword length and generator polynomial $g(x)$ blindly with the only knowledge of the received streams. For an encoding system, $m(x)$ is different in each codeword, but $g(x)$ is the same. According to Equations 1 and 2, the roots of $g(x)$ are also the roots of $c(x)$. If no error occurs, the roots of $g(x)$ will appear in every codeword. However, for an invalid codeword, this algebraic relationship does not exist. In this paper, we define the code roots as the roots of the generator polynomial. The root space of a binary codeword polynomial $c(x)$ defined in GF(2^m) ($m \geq 1$) is a finite space, which contains $2^m - 1$ symbols. We define **A** to be the set of the generator polynomial roots. In a noisy context, statistically, for each codeword $c(x)$, the probabilities of the codeword polynomial roots appear in **A** is larger than that in $\bar{A}$ (defined in GF(2^m)). While for an invalid codeword polynomial $c'(x)$, the roots of $c'(x)$ appear randomly in GF(2^m). In this case, the authors of [16-18] propose the following unproved hypothesis:

> Hypothesis 1: Each symbol in GF(2^m) has a uniform probability of being a root of $c'(x)$.

According to this hypothesis, the authors of [16-18] propose an algorithm to recognize the BCH code length by traversing all the possible code length and primitive polynomials to find the correct coding parameters that maximize the roots Information Dispersion Entropy Function (IDEF) as follows:

$$\begin{aligned}\Delta H &= -\sum_{i=1}^{n}\frac{1}{n}\log\frac{1}{n} - \left(-\sum_{i=1}^{n} p_i \log p_i\right) \\ &= \sum_{i=1}^{n} p_i \log p_i + \log n \end{aligned} \tag{3}$$

where $n = 2^m - 1$ is the code length, p_i ($1 \leq i \leq 2^m - 1$) is the probability of α^i to be the root of the code and α is a primitive element in GF(2^m). p_i is calculated as follows:

$$p_i = \frac{N_i}{N}, 1 \leq i \leq 2^m - 1. \tag{4}$$

The received sequence, i.e. R_c in Figure 1, is separated to M packets with an assumption of code length l, as shown in Figure 2. In [16-18], the authors assume that the start point of the first coding packet is obtained according to the frame synchronization testing, while the code length and generator polynomial are unknown. We define $r_j(x)(1 \leq j \leq M)$ to be the codeword polynomial of the jth packet in the received sequence. In Equation 4, N_i is the times of appearances of α^i being the root of $r_j(x)$ in the M packets, and $N = \sum_{i=1}^{2^m-1} N_i$.

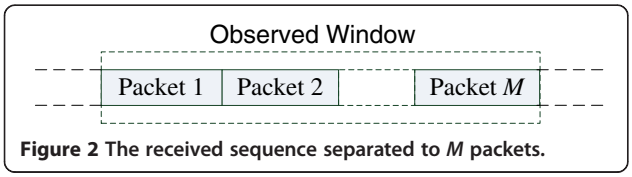

Figure 2 The received sequence separated to *M* packets.

According to Hypothesis 1, when the estimation of code length and primitive polynomial is incorrect, p_i could be considered uniformly distributed, and $p_i \approx 1/(2^m - 1)$ ($1 \leq i \leq 2^m - 1$). Thus the ΔH in Equation 2 is low. If the code parameters are estimated correctly and α^i is a root of $g(x)$, p_i should be larger. Therefore, the distribution of p_i should not be uniform. Then the information entropy of p_i is lower and ΔH is larger. This is the basic principle of estimating the code length by maximizing the ΔH defined in Equation 3.

Once the code length is estimated, by comparing p_i at different roots, we can consider the obviously higher ones as the estimation of the code roots and the generator polynomial could be obtained by $g(x) = (x-\alpha^{i_1})(x-\alpha^{i_2})\cdots(x-\alpha^{i_r})$, where α^{i_1}, α^{i_2}, $\cdots$, α^{i_r} are the estimated code roots, i.e. the roots of the generator polynomial.

The RIDERS algorithm has a good performance but there are still some drawbacks which need to be improved, which are described as follows:

1) Hypothesis 1 proposed in [16-18] is not correct. In section 3.2, we give the proof. In fact, not all the symbols in GF(2^m) have the same probability of being a root of an invalid codeword $c'(x)$.
2) This algorithm only considers the BCH codes in the cases of regular code length, i.e. code length $l = 2^m - 1$. The authors ignored the shortened code case, which are widely applied, however.
3) The code roots can be separated into some conjugate root groups, and each group contains several conjugate roots, which are the roots of a same minimal polynomial. If a generator polynomial $g(x)$ has a root β, which is a root of the minimal polynomial $m_p(x)$, the symbols which are other roots of $m_p(x)$ also are part of the roots of $g(x)$. So we can test which minimal polynomials are factors of the generator polynomial rather than testing which elements in GF(2^m) are roots of the code.
4) This algorithm is based on the hard decision symbols and do not utilize the soft channel outputs
5) This algorithm only considers the recognition of BCH codes and does not discuss the applications on other binary cyclic codes.
6) The authors of [16-18] ignore the synchronization of the codewords. They assume that the starting

positions of the codewords have been known before the recognition procedure by framing testing. But in practical implementations, this should not be the case in blind context.

In the paragraph from section 4, we propose an improved RIDERS algorithm based on soft-decision situations and extend the applications to general binary cyclic codes.

3.2 Proof of faultiness of Hypothesis 1

In this section, we present that Hypothesis 1 proposed in [16-18] is not always correct. The proof is shown below.

Proof. Let $c'(x)$ be the codeword polynomial of a codeword C', we can calculate p_i, which is the probability that α^i is a root of $c'(x)$. To calculate p_i, we define the minimal parity-check matrix $H_{\min}(\alpha^i)$ corresponding to the element α^i in GF(2^m) as follows:

$$H_{\min}(\alpha^i) = \left((\alpha^i)^{l-1}, (\alpha^i)^{l-2}, \cdots, (\alpha^i)^1, (\alpha^i)^0\right). \quad (5)$$

We transform $H_{\min}(\alpha^i)$ to its binary form by replacing the symbols in $H_{\min}(\alpha^i)$ by their binary column vector patterns according to the coding theory [25] and record it $Hb_{\min}(\alpha^i)$.

For example, the minimal parity-check matrix $H_{\min}(\alpha^3)$ corresponding to the element α^3 in GF(2^3) with code length $l = 2^3 - 1 = 7$ is as follows:

$$H_{\min}(\alpha^3) = \begin{pmatrix} \alpha^{18} & \alpha^{15} & \cdots & \alpha^3 & 1 \end{pmatrix}. \quad (6)$$

Based on the primitive polynomial $p(x) = x^3 + x + 1$, we can replace the symbol α^3 by the vector $[011]^T$, and other symbols are processed similarly. Then the parity-check matrix can be written in GF(2) as follows:

$$Hb_{\min}(\alpha^3) = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}. \quad (7)$$

If α^i is a root of $c'(x)$, we have

$$Hb_{\min}(\alpha^i) \times C' = 0 \quad (8)$$

There are m rows in $Hb_{\min}(\alpha^i)$ and we define $\mathbf{h}_\mu (1 \le \mu \le m)$ to be the μth row of $Hb_{\min}(\alpha^i)$. Then the equation $Hb_{\min}(\alpha^i) \times C' = 0$ means that the product of any row of $Hb_{\min}(\alpha^i)$ with the codeword C' equals to zero, as shown in Equation 9:

$$Hb_{\min}(\alpha^i) \times C' = 0 \Leftrightarrow \begin{cases} \mathbf{h}_1 \times C' = 0 \\ \mathbf{h}_2 \times C' = 0 \\ \vdots \\ \mathbf{h}_m \times C' = 0 \end{cases} \quad (9)$$

So we can calculate the probability of α^i being a root of $c'(x)$, i.e. the probability of $Hb_{\min}(\alpha^i) \times C' = 0$ as follows:

$$P_r\left[Hb_{\min}(\alpha^i) \times C' = 0\right] = P_r(\mathbf{h}_1 \times C' = 0, \mathbf{h}_2 \times C' = 0, \ldots, \mathbf{h}_m \times C' = 0) \quad (10)$$

In the following paragraphs of this paper, we define $P_r(\mathbf{x})$ as the probability of $\mathbf{x}$. Let $h_{\mu,u} (1 \le u \le n)$ and C_u be the uth elements in the vector $\boldsymbol{h}_\mu$ and C' and we define the *checking indexing set* $\mathbf{S}_\mu$ for h_μ and C' as follows:

$$S_\mu = \{C_u | h_{\mu,u} = 1\} \quad (11)$$

Obviously, when the number of nonzero elements in $\mathbf{S}_\mu$ is even, we have

$$\mathbf{h}_\mu \times C' = 0 \quad (12)$$

And when the number of nonzero elements in $\mathbf{S}_\mu$ is odd, we have

$$\mathbf{h}_\mu \times C' = 1 \quad (13)$$

When C' is not a valid codeword, i.e. the elements in C' can be considered to appear randomly, the probabilities of the number of nonzero elements in $\mathbf{S}_\mu$ being odd and even are all about 0.5. When $Hb_{\min}(\alpha^i)$ is full rank (the rank is calculated in GF(2)), the rows of $Hb_{\min}(\alpha^i)$ is linearly independent, so we can calculate Equation 10 as follows:

$$P_r\left[Hb_{\min}(\alpha^i) \times C = 0\right] = \prod_{\mu=1}^{m} P_r(\mathbf{h}_\mu \times C = 0) = (0.5)^m \quad (14)$$

But if $Hb_{\min}(\alpha^i)$ is not full rank, the calculation of $P_r[Hb_{\min}(\alpha^i) \times C = 0]$ by Equation 14 is not correct. We define the *maximum linearly independent vector group* **MI** of the row vectors set $H = \{\boldsymbol{h}_\mu | 1 \le \mu \le m\}$ as follows:

MI is a subset of **H** and meets the following conditions:

(1) The vectors in MI are linearly independent;
(2) Any vector in H can be obtained by linear combinations of the vectors in MI.

And it is easy to prove that the number of vectors in **MI** equals to the rank of $Hb_{\min}(\alpha^i)$.

According to the condition 2 of the definition of **MI**, if all the vectors in $\{\boldsymbol{h}_\mu | \boldsymbol{h}_\mu \in \mathbf{MI}\}$ make $\boldsymbol{h}_\mu \times C = 0$, then

also for all the vectors in $\{\boldsymbol{h}_\mu|\boldsymbol{h}_\mu \in \mathbf{H}\}$, we have $\boldsymbol{h}_\mu \times C = 0$. So the calculation of Equation 10 should be:

$$\begin{aligned} &P_r\left[Hb_{\min}(\alpha^i) \times C = 0\right] \\ &= \prod_{\theta=1}^{\operatorname{rank}(Hb_{\min}(\alpha^i))} P_r\left(\mathbf{h}_{\mu_\theta} \times C = 0\right) \\ &= (0.5)^{\operatorname{rank}(Hb_{\min}(\alpha^i))}, \end{aligned} \tag{15}$$

where the elements in $\left\{\mathbf{h}_{\mu_\theta}|1\le\theta\le\operatorname{rank}(Hb_{\min}(\alpha^i))\right\}$ are the vectors in **MI**, i.e. a maximum linearly independent vector group of the rows of $Hb_{\min}(\alpha^i)$.

According to Equation 15, Hypothesis 1 is true only if all the $Hb_{\min}(\alpha^i)$, where $1 \le i \le 2^m - 1$, have the same rank. Unfortunately, this condition cannot always be met. For example, we have the following results over GF(2^6):

$$\begin{cases} \operatorname{rank}((Hb_{\min}(\alpha^1))) = 6 \\ \operatorname{rank}(Hb_{\min}(\alpha^{21})) = 2 \\ \operatorname{rank}(Hb_{\min}(\alpha^{63})) = 1 \\ \cdots \end{cases} \tag{16}$$

Therefore, we have

$$\begin{cases} P_r[Hb_{\min}(\alpha^1) \times C = 0] = \left(\frac{1}{2}\right)^6 \\ P_r[Hb_{\min}(\alpha^{21}) \times C = 0] = \left(\frac{1}{2}\right)^2 \\ P_r[Hb_{\min}(\alpha^{63}) \times C = 0] = \left(\frac{1}{2}\right)^1 \\ \cdots \end{cases} \tag{17}$$

Therefore, we can get the conclusion that Hypothesis 1 proposed in [16-18] is not correct.

Figure 3 shows the probabilities that the elements in GF(2^6) are the roots of a random block with length l = 63 by simulations.

4. Blind recognition algorithm in soft-decision situations

4.1 Code length estimation and blind block synchronization

Soft outputs of the channel could provide more information about the reliability of each decision symbol. In this section, we propose an approach to improve the recognition performance by employing the soft decisions.

We define $c_r(x)$ to be the codeword polynomial of a code block C_r. According to the algebraic principles of cyclic codes, if α^i is a root of $c_r(x)$, we have $c_r(\alpha^i) = 0$ and $H_{\min}(\alpha^i) \times C_r = 0$. In soft-decision situations, instead of verifying whether α^i is a root of each block, we can calculate $p_{j,i}$, the probability that α^i is a root of the jth block in the received sequence as shown in Figure 2, and calculate p_i in Equation 4 as follows:

$$p_i = \frac{\sum_{j=1}^{M} p_{j,i}}{\sum_{i=1}^{2^m-1}\sum_{j=1}^{M} p_{j,i}}, 1 \le i \le 2^m-1, \tag{18}$$

where M is the number of blocks, as shown in Figure 2.

The elements in an extension field GF(2^m) can be separated to some groups according to the minimal elements over GF(2^m). Each minimal polynomial has several roots in GF(2^m), we call the set of them as a conjugate element group in this paper. And the generator polynomial of a cyclic code can be factorized by some minimal polynomials as follows:

$$g(x) = m_1(x)m_2(x)...m_p(x) \tag{19}$$

Because the generator polynomial $g(x)$ is a factor of a valid codeword polynomial $c(x)$, the minimal polynomials in Equation 19 are also the factors of $c(x)$. So if an element $\alpha^i(1 \le i \le 2^m - 1)$ in GF(2^m) is a root of $c(x)$, the elements which have the same minimal polynomial with α^i are also the roots of $c(x)$. Therefore, we can just calculate $p'_\lambda(1 \le \lambda \le q)$, the probability that the minimal polynomial $m_\lambda(x)(1 \le \lambda \le q)$ is a factor of $c_r(x)$, where q denotes the number of minimal polynomials over GF (2^m). According to this idea, we can modify Equation 18 to Equation 20 to calculate p'_λ rather than p_i. This modification can reduce the calculation complexity because the number of minimal polynomials over GF(2^m) is severely lower than the number of elements in GF (2^m). In Equation 20, $p'_{j,\lambda}$ denotes the probability that $m_\lambda(x)$ is a factor of the codeword polynomial of the jth block in the observed window as shown in Figure 2.

$$p'_\lambda = \frac{\sum_{j=1}^{M} p'_{j,\lambda}}{\sum_{\lambda=1}^{q}\sum_{j=1}^{M} p'_{j,\lambda}}, 1 \le \lambda \le q \tag{20}$$

And the IDEF defined in Equation 3 should be modified to Equation 21:

$$\begin{aligned} \Delta H &= -\sum_{\lambda=1}^{q}\frac{1}{q}\log\frac{1}{q} - \left(-\sum_{\lambda=1}^{q} p'_\lambda \log p'_\lambda\right) \\ &= \sum_{\lambda=1}^{q} p'_\lambda \log p'_\lambda + \log q \end{aligned} \tag{21}$$

To calculate $p'_{j,\lambda}$ in Equation 20, which is the probability that a minimal polynomial $m_\lambda(x)$ is a factor of $c_r(x)$, we can define the binary minimal parity-check

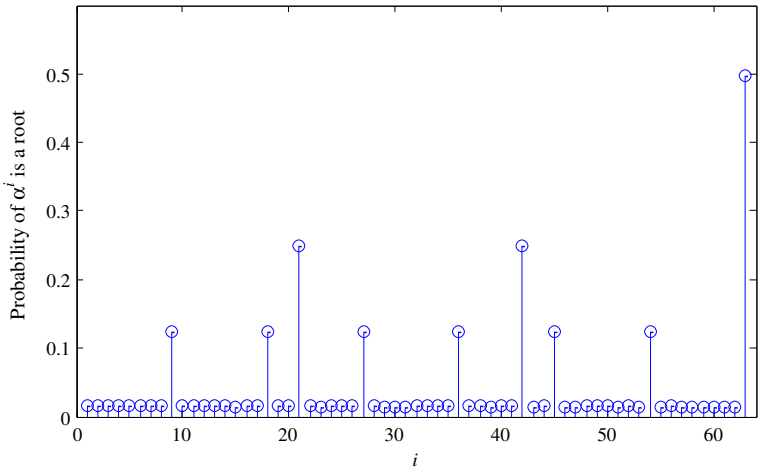

Figure 3 Probability of the elements in GF(2^6) being the roots of random codes.

matrix $Hb_{\min}(m_\lambda(x))$ corresponding to $m_\lambda(x)$ and calculate the probability of $Hb_{\min}(m_\lambda(x)) \times C_r = 0$.

The coefficients of $m_\lambda(x)$ are in GF(2) and $m_\lambda(x)$ can be written as follows:

$$m_\lambda(x) = g_e x^e + g_{e-1} x^{e-1} + \cdots + g_1 x + g_0 \tag{22}$$

where e is the degree of $m_\lambda(x)$. g_e, g_{e-1}, $\cdots$, g_1 and g_0 are all in GF(2). According to these coefficients of $m_\lambda(x)$, we can obtain the minimal polynomial-based binary, minimal parity-check matrix $Hb_{\min}(m_\lambda(x))$ with the following steps.

1) We assume the code length is l and initialize a matrix G as follows:
In Equation 23, the number of rows and columns are l-e and l, respectively.

$$G = \begin{pmatrix} g_e & g_{e-1} & \cdots & g_1 & g_0 & 0 & \cdots & \\ 0 & g_e & g_{e-1} & \cdots & g_1 & g_0 & 0 & \cdots \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ & \cdots & 0 & g_e & g_{e-1} & \cdots & g_1 & g_0 \end{pmatrix} \tag{23}$$

2) Transform the left $e \times e$ area of G to an identity matrix I by elementary row transformation as follows:
where Q is a matrix, which has l-e rows and e columns.

$$G = (I|Q), \tag{24}$$

3) The minimal parity-check matrix can be obtained as follows:

$$Hb_{\min}(m_\lambda(x)) = \left(Q^T|I\right) \tag{25}$$

According to the algebraic principles of coding theories, we can calculate the syndromes corresponding to $Hb_{\min}(m_\lambda(x))$ by Equation 25 [23]:

$$\begin{aligned} S &= [S(1), S(2), \cdots, S(n_r)]^T \\ &= Hb_{\min}(m_\lambda(x)) \times C_r, \end{aligned} \tag{26}$$

where n_r is the number of rows in $Hb_{\min}(m_\lambda(x))$, i.e. the degree of $m_\lambda(x)$. If $m_\lambda(x)$ is a factor of $c_r(x)$ and no error occurs during the transmission, all syndromes should equal to zero. If the block contains errors or $m_\lambda(x)$ is not a factor of $c_r(x)$, not all the syndromes equal to zero. So when the minimal polynomials, which are the factors of the generator polynomial, are correctly estimated, the probability of $\boldsymbol{S} = 0$ is larger than the case of incorrect estimation of the minimal polynomials. $p'_{j,\lambda}$ in Equation 20 can be calculated as follows:

$$p'_{j,\lambda} = \frac{1}{n_r} \sum_{k=1}^{n_r} P_r[S_H(k) = 0], 1 \le k \le n_r, \tag{27}$$

where $P_r[S_H(k) = 0][1 \le k \le n_r]$ is the probability of $S_H(k) = 0(1 \le k \le n_r)$, k denotes the corresponding row number of $Hb_{\min}(m_\lambda(x))$. In fact, $p'_{j,\lambda}$ calculated in Equation 27 is not the probability that $m_\lambda(x)$ is a factor of the codeword polynomial, it is just the mean value of the probabilities that the syndromes equal to zero. The true probability should be obtained by calculating the probability that all syndromes equal to zero. But as shown in section 3.2, the probability that all syndromes equal to zero is determined by the degree of the corresponding minimal polynomial for incorrect coding parameter estimations, the probability distribution is not uniform. But we use the mean value of $P_r[S_H(k) = 0]$ to indirectly depict the probability that a minimal polynomial is a factor of the codeword polynomial, the influence of the degree of the difference minimal polynomials is low. In this case, we can assume that for a

random data, the distribution of the probabilities of the minimal polynomials being factors of the codeword polynomials is approximately uniform.

Jing proposed the Adaptive Belief Propagation (ABP) method on soft-Input Soft-Output decoding of RS codes [26]. The main idea is adapting the parity-check matrix of the codes to the reliability of the received information bits at each iteration step of the iterative decoding procedure. This idea is also employed in [22] to achieve blind frame synchronization of RS codes. The adaptation procedure reduces the impact of most unreliable decision bits on the calculation of syndromes. In our work, we also utilize the adaptation algorithm introduced in [23] and [26] before using Equation 27. The adaptive processing for a given received codeword C_r and a binary minimal parity-check matrix $Hb_{\min}(m_\lambda(x))$ includes the following steps:

1) Combine $Hb_{\min}(m_\lambda(x))$ and C_r^T to form a matrix $H^*(m_\lambda(x))$ as follows:

$$H^*(m_\lambda(x)) = \left[\begin{array}{cccc} r_1 & r_2 & \cdots & r_l \\ \hline h_{1,1} & h_{1,2} & \cdots & h_{1,l} \\ h_{2,1} & h_{2,2} & \cdots & h_{2,l} \\ \vdots & \vdots & \ddots & \vdots \\ h_{n_r,1} & h_{n_r,2} & \cdots & h_{n_r,l} \end{array}\right] \tag{28}$$

where $r_1, r_2, \ldots, r_3$ are the soft-decision bits of the codeword C_r, $\{h_{k,u} | 1 \le k \le n_r, 1 \le u \le l\}$ are the elements of $Hb_{\min}(m_\lambda(x))$ in GF(2).

2) Replace each r_u $(1 \le u \le l)$ in $H^*(m_\lambda(x))$ with their absolute values to form a new matrix $H_r^*(m_\lambda(x))$, adjust the positions of the columns in $H_r^*(m_\lambda(x))$ to make the first row in $H_r^*(m_\lambda(x))$ is ranked from the lowest to the highest and record the indexes. The absolute values of $\{r_u | 1 \le u \le l\}$ denote the reliabilities of the received soft-decision bits. As shown in Equation 29, $|r_{i_1}| \le |r_{i_2}| \le \cdots \le |r_{i_l}|$ and $i_1, i_2, \cdots, i_l$ are the column indexes of $r_{i_1}, r_{i_2}, \cdots, r_{i_l}$ in $H^*(m_\lambda(x))$.

$$H_r^{\ *}(m_\lambda(x)) = \left[\begin{array}{cccc} |r_{i_1}| & |r_{i_2}| & \cdots & |r_{i_l}| \\ \hline h_{1,i_1} & h_{1,i_2} & \cdots & h_{1,i_l} \\ h_{2,i_1} & h_{2,i_2} & \cdots & h_{2,i_l} \\ \vdots & \vdots & \ddots & \vdots \\ h_{n_r,i_1} & h_{n_r,i_2} & \cdots & h_{n_r,i_l} \end{array}\right] \tag{29}$$

3) Transform $H_r^*(m_\lambda(x))$ by elementary row operations to make the last n_r elements of the first column in $H_r^*(m_\lambda(x))$ has only one "1" at the top, as shown in Equation 30. The first row does not join the elementary transformations.

$$H_r^{\ *}(m_\lambda(x)) = \left[\begin{array}{cccc} |r_{i_1}| & |r_{i_2}| & \cdots & |r_{i_l}| \\ \hline 1 & x & \cdots & x \\ 0 & x & \cdots & x \\ \vdots & \vdots & \ddots & \vdots \\ 0 & x & \cdots & x \end{array}\right] \tag{30}$$

This transformation limits the influences of the most unreliable decision bit to only one syndrome element. Furthermore, we continue the elementary transformation on $H_r^*(m_i(x))$ to limit the numbers of "1" in the following n_r-1 columns to one (except the first row), as shown in Equation 31.

$$H_r^*(m_\lambda(x)) = \left[\begin{array}{ccccccccc} |r_{i_1}| & |r_{i_2}| & |r_{i_3}| & \cdots & |r_{i_{n_r}}| & |r_{i_{n_r+1}}| & \cdots & |r_{i_{l-1}}| & |r_{i_l}| \\ 1 & 0 & 0 & \cdots & 0 & x & \cdots & x & x \\ 0 & 1 & 0 & \cdots & 0 & x & \cdots & x & x \\ 0 & 0 & 1 & \cdots & 0 & x & \cdots & x & x \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & x & x \\ 0 & 0 & 0 & 0 & 1 & x & \cdots & x & x \end{array}\right] \tag{31}$$

When the left bottom $n_r \times n_r$ area becomes an indent matrix, stop the operation. Then the last n_r rows in $H_r^*(m_\lambda(x))$ form a new matrix. We recover its original column orders and call it $Hb_{\min\_a}(m_\lambda(x))$. Because the transformation is elementary, the relationship $Hb_{\min\_a}(m_\lambda(x)) \times C_r = 0$ in the hard decision situations still exists if C_r is a valid codeword. So we can calculate the probability $P_r[S_H(k) = 0]$ according to $Hb_{\min\_a}(m_\lambda(x))$. This replacement reduces the influences of the n_r most unreliable decision bits.

In this paper, we assume that the transmitter is sending a binary sequence of codewords and using a binary phase shift keying (BPSK) modulation, i.e. let +1 and −1 be the modulated symbols of 0 and 1. The modulation operation from code bit c to modulated symbol s could be written as $s = 1 - 2c$, and we assume that the propagation channel is a binary symmetry channel which is corrupted by an additive white Gaussian noise (AWGN). For each configuration, the information symbols in the codes are randomly chosen. A received symbol r could be expressed as $r = s + w$, where w is the AWGN.

According to the previous assumptions, s is an equally probable binary random variable and

$$P_r(s=+1)=P_r(s=-1)=1/2 \tag{32}$$

The noise w follows a normal distribution with the probability density function (PDF)

$$f(x)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{x^2}{2\sigma^2}\right) \tag{33}$$

So the conditional PDF of r is

$$f\left(r\middle|s\right)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-s)^2}{2\sigma^2}\right) \tag{34}$$

where $\sigma^2=\frac{1}{2(E_s/N_0)}$ is the variance of the noise.

For a given received bit r, we can obtain the following conditional probabilities:

$$\begin{aligned}&P_r(s=+1|r)\\&=\frac{f(r|s=+1)\times P_r(s=+1)}{f(r|s=+1)\times P_r(s=+1)+f(r|s=-1)\times P_r(s=-1)}\\&=\frac{\exp(2r/\sigma^2)}{1+\exp(2r/\sigma^2)}\end{aligned} \tag{35}$$

$$\begin{aligned}P_r(s=-1|r)&=1-P_r(s=+1|r)\\&=\frac{1}{1+\exp(2r/\sigma^2)}\end{aligned} \tag{36}$$

Let $\boldsymbol{r}=[r_1, r_2, \ldots, r_n, r_{n+1}, \ldots]$ be a received soft-decision vector corresponding to the random modulated vector $s=[s_1, s_2, \ldots, s_n, s_{n+1}, \ldots]$. We now calculate the conditional probabilities of $s_1 \oplus s_2 = +1$ and $s_1 \oplus s_2 = -1$. According to the mapping operation defined by $s = 1-2c$, we have

$$\begin{aligned}&P_r(s_1\oplus s_2=+1|\boldsymbol{r})=P_r(s_1=+1|r_1)\times P_r(s_2=+1|r_2)\\&\quad+P_r(s_1=-1|r_1)\\&\quad\times P_r(s_2=-1|r_2)\\&=\frac{1}{2}+\frac{1}{2}\prod_{u=1}^{2}\frac{\exp(2r_u/\sigma^2)-1}{\exp(2r_u/\sigma^2)+1}\end{aligned} \tag{37}$$

$$\begin{aligned}P_r(s_1\oplus s_2=-1|\boldsymbol{r})&=1-P_r(s_1\oplus s_2=+1|\boldsymbol{r})\\&=\frac{1}{2}-\frac{1}{2}\prod_{u=1}^{2}\frac{\exp(2r_u/\sigma^2)-1}{\exp(2r_u/\sigma^2)+1}\end{aligned} \tag{38}$$

Similarly, we can calculate the conditional probabilities of $s_1 \oplus s_2 \oplus s_3 = +1$ and $s_1 \oplus s_2 \oplus s_3 = -1$ as follow:

$$\begin{aligned}&P_r(s_1\oplus s_2\oplus s_3=+1|\boldsymbol{r})=P_r(s_1\oplus s_2=+1|\boldsymbol{r})\\&\quad\times P_r(s_3=+1|r_3)+P_r(s_1\oplus s_2=-1|\boldsymbol{r})\\&\quad\times P_r(s_3=-1|r_3)=\frac{1}{2}+\frac{1}{2}\prod_{u=1}^{3}\frac{\exp(2r_u/\sigma^2)-1}{\exp(2r_u/\sigma^2)+1}\end{aligned} \tag{39}$$

$$\begin{aligned}P_r(s_1\oplus s_2\oplus s_3=-1|\boldsymbol{r})&=1-P_r(s_1\oplus s_2\oplus s_3=+1|\boldsymbol{r})\\&=\frac{1}{2}-\frac{1}{2}\prod_{u=1}^{3}\frac{\exp(2r_u/\sigma^2)-1}{\exp(2r_u/\sigma^2)+1}\end{aligned} \tag{40}$$

We define the XOR-SUM operation as $\sum_{u=1}^{n}\otimes s_u = s_1\oplus s_2\oplus\cdots\oplus s_n$ and assume that the conditional probabilities of XOR-SUM can be expressed as Equation 41:

$$\begin{cases}P_r\left(\sum_{u=1}^{n}\otimes s_u=+1|\boldsymbol{r}\right)=\frac{1}{2}+\frac{1}{2}\prod_{u=1}^{n}\frac{\exp(2r_u/\sigma^2)-1}{\exp(2r_u/\sigma^2)+1}\\P_r\left(\sum_{u=1}^{n}\otimes s_u=-1|\boldsymbol{r}\right)=\frac{1}{2}-\frac{1}{2}\prod_{u=1}^{n}\frac{\exp(2r_u/\sigma^2)-1}{\exp(2r_u/\sigma^2)+1}\end{cases} \tag{41}$$

Then, we have

$$\begin{aligned}&P_r\left(\sum_{u=1}^{n+1}\otimes s_u=+1|\boldsymbol{r}\right)=P_r\left(\sum_{u=1}^{n}\otimes s_u=+1|\boldsymbol{r}\right)\\&\quad\times P_r(s_{n+1}=+1|\boldsymbol{r})+P_r\left(\sum_{u=1}^{n}\otimes s_u=-1|\boldsymbol{r}\right)\\&\quad\times P_r(s_{n+1}=-1|\boldsymbol{r})=\frac{1}{2}+\frac{1}{2}\prod_{u=1}^{n+1}\frac{\exp(2r_u/\sigma^2)-1}{\exp(2r_u/\sigma^2)+1}\\&P_r\left(\sum_{u=1}^{n+1}\otimes s_u=-1|\boldsymbol{r}\right)=1-P_r\left(\sum_{u=1}^{n+1}\otimes s_u=+1|\boldsymbol{r}\right)\\&=\frac{1}{2}-\frac{1}{2}\prod_{u=1}^{n+1}\frac{\exp(2r_u/\sigma^2)-1}{\exp(2r_u/\sigma^2)+1}\end{aligned} \tag{42, 43}$$

According to the induction principle, the expression of the conditional probabilities in Equation 41 turns out to be true, and could be simplified as follows:

$$\begin{cases}P_r\left(\sum_{u=1}^{n}\otimes s_u=+1|\boldsymbol{r}\right)=\frac{1}{2}+\frac{1}{2}\prod_{i=1}^{n}\tanh\left(r_u/\sigma^2\right)\\P_r\left(\sum_{u=1}^{n}\otimes s_u=-1|\boldsymbol{r}\right)=\frac{1}{2}-\frac{1}{2}\prod_{i=1}^{n}\tanh\left(r_u/\sigma^2\right)\end{cases} \tag{44}$$

By employing Equation 44, we can calculate the probability $P_r[S_H(k) = 0]$ as follows:

$$P_r[S_H(k)=0] = P_r\left(\sum_{v=1}^{w_k} \oplus s_{u_v} = +1|\boldsymbol{r}\right) = \frac{1}{2} + \frac{1}{2}\prod_{v=1}^{w_k} \tanh\left(r_{u_v}/\sigma^2\right), \tag{45}$$

where w_k is the number of ones in the kth row of the adapted minimal binary parity-check matrix $Hb_{\min\_a}(m_\lambda(x))$, u_v represents the position of the vth non-zero element in the kth row of $Hb_{\min\_a}(m_\lambda(x))$. s_{u_v} and r_{u_v} are the u_vth modulated symbol on the transmitter and the corresponding soft-decision output on the receiver, respectively.

In shortened code cases, a codeword with block length l and shortened length l_s can be obtained by choosing the last l elements from a codeword which has a regular length $(l + l_s)$ as follows:

$$C_w = \left(\underbrace{0 \quad \cdots \quad 0}_{l_s \text{zeros}} \underbrace{c_l \quad c_{l-1} \quad \cdots \quad c_0}_{l \text{ elements}}\right), \tag{46}$$

where the first l_s elements of C_w are zeros. Therefore, we can simply obtain the minimal parity-check matrices of the shortened codes by deleting the first l_s columns of $Hb_{\min}(m_\lambda(x))$.

4.2 Recognition of generator polynomials

After the code length and synchronization position estimation, the extension field degree m corresponding to the being recognized code can also be obtained. Then we can list the minimal polynomials over GF(2^m) and find out which ones are factors of the generator polynomial. These minimal polynomials can also be recognized according to the probabilities of syndromes equaling to zero.

In the procedure of the code length and synchronization position estimation, we have calculated the probability that a minimal polynomial is a factor of the received codeword polynomials. We assume that the estimated code length and extension field degree are l and m, the number of minimal polynomials over GF(2^m) is q and $m_1(x)$, $m_2(x)$, ..., $m_q(x)$ are the minimal polynomials over GF(2^m).

According to Equation 45, we can calculate the kth syndrome for a given minimal parity-check matrix of $Hb_{\min}(m_\lambda(x))$. Equation 47 is the log-likelihood ratios (LLR) of $P_r[S_H(k) = 0]$, where $H = Hb_{\min}(m_\lambda(x))$ is

$$L[S_H(k)] = \log\frac{P_r[S_H(k)=0]}{P_r[S_H(k)\neq 0]} = \log\frac{1+\prod_{v=1}^{w_k}\tanh\left(r_{u_v}/\sigma^2\right)}{1-\prod_{v=1}^{w_k}\tanh\left(r_{u_v}/\sigma^2\right)} = 2\text{artanh}\left[\prod_{u=1}^{w_k}\tanh\left(r_{u_v}/\sigma^2\right)\right] \tag{47}$$

And we propose to calculate a likelihood criterion (LC) of $m_\lambda(x)(1 \le i \le q)$ being a factor of the generator polynomial as follows:

$$L(m_\lambda(x)) = \sum_{j=1}^{M}\frac{1}{n_r}\sum_{k=1}^{n_r} L_j\left[S_{Hb_{\min_a}(m_\lambda(x))}(k)\right], \quad 1\le\lambda\le q, \tag{48}$$

where $Hb_{\min\_a}(m_\lambda(x))$ is the adapted minimal parity-check matrix corresponding to the minimal polynomial $m_\lambda(x)$, M is the number of packets in the observed window W as shown in Figure 2, n_r is the number of the rows in $Hb_{\min\_a}(m_\lambda(x))$, $L_j\left[S_{Hb_{\min\_a}(m_\lambda(x))}(k)\right]$ is the LLR defined by Equation 47 and calculated at the jth block of the observed window W. According to Equation 48, we can calculate the LCs of all the minimal polynomials over GF(2^m). By comparing the LCs, we can choose the minimal polynomials, LCs of which are obviously higher than others, as the estimated factors of the generator polynomial, then the generator polynomial is obtained.

However, we can test whether the product of several most likely minimal polynomials is a factor of the generator polynomial to increase the successful recognition rate, because according to the adaptive processing of the parity-check matrices, the more parity equations we consider, the more we are able to construct a parity matrix which is parsed on less reliable bits. For the convenience of automatic recognition using computer programs, we propose the procedure including the following steps to estimate the optimal parity-check matrix:

Step 1: Calculate the LCs to form a vector L:

$$L = \left[L(m_1(x)), L(m_2(x)), \cdots, L\left(m_q(x)\right)\right] \tag{49}$$

Step 2: Rank the vector L from the highest to the lowest, in order to form a new vector L_R as follows:

$$L_R = \left[L(m_{\lambda_1}(x)), L(m_{\lambda_2}(x)), \cdots, L\left(m_{\lambda_q}(x)\right)\right] \tag{50}$$

and record the indexes:

$$I = \left[\lambda_1, \lambda_2, \cdots, \lambda_q\right] \tag{51}$$

where $\lambda_\omega(1 \le \omega \le q)$ denotes the index of $L(m_{\lambda_\omega}(x))$ in L.

Step 3: Let ω increase from 1 to q, combine the binary minimal parity matrices for the minimal polynomials $m_{\lambda_1}(x)...m_{\lambda_\omega}(x)$, in order to form H_ω as follows:

$$H_\omega = \begin{pmatrix} Hb_{\min}(m_{\lambda_1}(x)) \\ Hb_{\min}(m_{\lambda_2}(x)) \\ \vdots \\ Hb_{\min}(m_{\lambda_\omega}(x)) \end{pmatrix}, \quad 1 \le \omega \le q \tag{52}$$

After adaptive processing for H_ω, calculate the LCs of $H_\omega \times C_r = 0 (1 \le \omega \le q)$ by Equation 53 and obtain the LC vector L_H as shown in Equation 54.

$$L(H_\omega) = \sum_{k=1}^{n_r} L[S_{H_\omega}(k)], \qquad 1 \le \omega \le q \tag{53}$$

$$L_H = \left[L(H_1), L(H_2), \cdots, L\left(H_q\right)\right] \tag{54}$$

Step 4: Find the maximal element of L_H, record the corresponding matrix $H\hat{\omega}$

Step 5: According to Equations 49 and 50, we can find the polynomials $m_{\lambda_1}(x) \quad \ldots \quad m_{\lambda\hat{\omega}}(x)$ and write the generator polynomial as follows:

$$g(x) = m_{\lambda_1}(x) m_{\lambda_2}(x) ... m_{\lambda\hat{\omega}}(x) \tag{55}$$

But in our work, we find that some minimal polynomials are easily lost. These minimal polynomials have the minimal parity-check matrices with low rows, so the adaptive processing can only reduce the influence for low number of unreliable decision bits. For example, consider the following minimal polynomials corresponding to the elements α^1, α^9 and α^0 in GF(2^6):

$$\begin{cases} m_1(x) = x^6 + x^1 + 1 \\ m_2(x) = x^3 + x^2 + 1 \\ m_3(x) = x + 1 \end{cases} \tag{56}$$

The degrees of $m_1(x)$, $m_2(x)$ and $m_3(x)$ are 6, 3, and 1, respectively. Therefore, the number of rows of the binary minimal parity-check matrices $Hb_{\min}(m_1(x))$, $Hb_{\min}(m_2(x))$ and $Hb_{\min}(m_3(x))$ corresponding to $m_1(x)$, $m_2(x)$ and $m_3(x)$ are also 6, 3, and 1, respectively. So $Hb_{\min}(m_1(x))$, $Hb_{\min}(m_2(x))$ and $Hb_{\min}(m_3(x))$ can limit the influences of 6, 3, 1 unreliable decision bits after adaptive processing, respectively. For $m_2(x)$ and $m_3(x)$, the LCs of $Hb_{\min\_a}(m_2(x))$ and $Hb_{\min\_a}(m_3(x))$, especially $Hb_{\min\_a}(m_2(x))$, may lower than the incorrect minimal polynomials when the signal-to-noise ratio (SNR) is low. In this case, the ranking of LCs in Equation 50 may not be correct, so the generator polynomial recognition is failed. To solve this problem, we can additionally combine these minimal parity-check matrices with $H\hat{\omega}$ obtained in Step 4 described previously and check whether the corresponding minimal polynomials are also factors of the generator polynomials. The details of the additional steps are listed below:

Step 6: List the binary minimal parity-check matrices over GF(2^m) which have low rows: $Hb_{\min}(m_{L1}(x))$, $Hb_{\min}(m_{L2}(x))$,..., $Hb_{\min}(m_{L\eta}(x))$, here η represents the number of binary minimal parity-check matrices with low rows.

Step 7 Record $\mathrm{LC}_{\max} = \mathrm{LC}(H\hat{\omega})$ and initialize a variable τ to be 1.

Step 8: Combine $H\hat{\omega}$ and $Hb_{\min}(m_{L\tau}(x))$ to form a new parity-check matrix $H_{\hat{\omega},\tau}$ as follow:

$$H_{\hat{\omega},\tau} = (H\hat{\omega}\, Hb_{\min}(m_{L\tau}(x))) \tag{57}$$

Step 9: If $\mathrm{LC}\left(H_{\hat{\omega},\tau}\right) > 0.9 \times \mathrm{LC}_{\max}$, let $H\hat{\omega} = H_{\hat{\omega},\tau}$ and $\mathrm{LC}_{\max} = \max\left(\mathrm{LC}_{\max}, \mathrm{LC}\left(H_{\hat{\omega},\tau}\right)\right)$.

Step 10: If $\tau = \eta$, execute step 11; else, let $\tau = \tau + 1$ and go back to step 8.

Step 11: Output the newly obtained $H\hat{\omega}$ as the final estimation of the parity-check matrix and get the generator polynomials according to the minimal polynomials corresponding to $H\hat{\omega}$.

5. General recognition procedure

In this section, we present the general procedure for the blind recognition of binary cyclic codes based on the principles proposed in the previous sections. Before the recognition, some prior information could help to estimate the possible range of the code length l. Then, we traverse all the possible values of code length l and codeword starting position t and choose the parameter pair (l, t) which maximizes the IDEF defined in Equation 21 to be the estimated code length and block synchronization position. Note that to get the minimal polynomials for each code length l over an extension field GF(2^m), we must know the field exponent m of the code. For an ordinary binary cyclic code, its code length is $2^m - 1$, while the code length l of a shortened code is $= 2^{2^m} - 1 - l_s$, where l_s is the shortened length. Therefore, the minimal value of the field exponent m for a code length l is the smallest integer k such that $< 2^{2^k}$. The maximal value of m should be estimated with some prior information. For each code length l and synchronization position t, we traverse all the possible extension field degrees to calculate ΔH, and choose the maximum one as $\Delta H(l,t)$. After the code length estimation, we search for the minimal polynomials which are the factors of the generator polynomial by the algorithm described in section 4.2.

The general recognition procedure is listed below:

Step 1: According to some prior information, set the searching range of the code length l, i.e. set the minimal and maximal code length $l_{\min}$ and $l_{\max}$.

Step 2: Design a window W which has a length L at least $10 \times l_{\max}$, i.e. $M \ge 10$ in Figure 2.

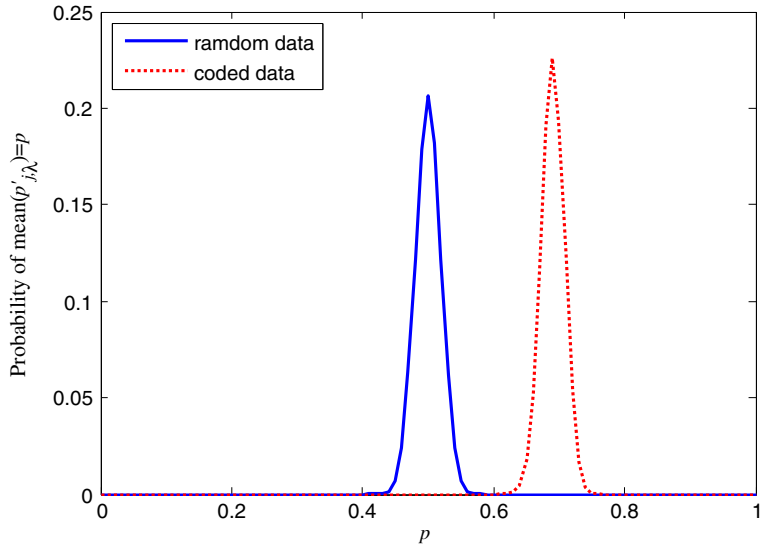

Figure 4 Probability density distribution of p'_{λ} for the coded and random data.

Step 3: Full fill the window W with the received soft-decision bits.
Step 4: Set the code length $l = l_{\min}$.
Step 5: Set the initial synchronization position t at 0, which is the starting position of W.
Step 6: Assume the code length is l and the synchronization position is t and calculate ΔH. Note that the window W has more than one assumed codewords, we calculate the ΔH on all the codewords and compute the mean of them as $\Delta H(l,t)$.
Step 7: If $t < l$, then let $t = t + 1$ and go back to step 6; if $t = l$, then jump to step 8.
Step 8: If $l < l_{max}$, then let $l = l + 1$ and go back to step 5; if $l = l_{max}$, then jump to step 9.
Step 9: Compare all the calculated $\Delta H(l,t)$, select the maximum one and get the corresponding values of l, t and m as the estimated code length, synchronization position and the degree of the GF of the recognized codes, respectively.
Step 10: Let the code length and synchronization position be the estimated parameters l and t, fetch M codewords from the observed window W. And list the minimal polynomials over GF(2^m), which are $m_1(x)$, $m_2(x)$,..., $m_q(x)$.

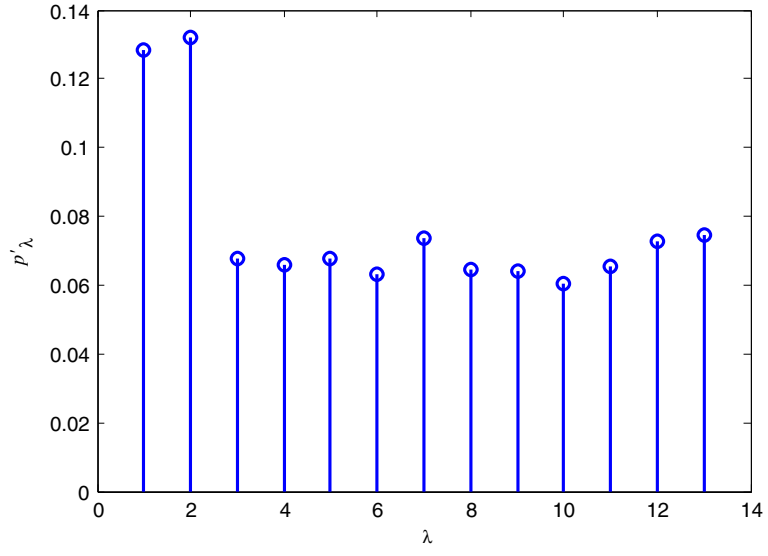

Figure 5 Values of p'_{λ} under correct parameters.

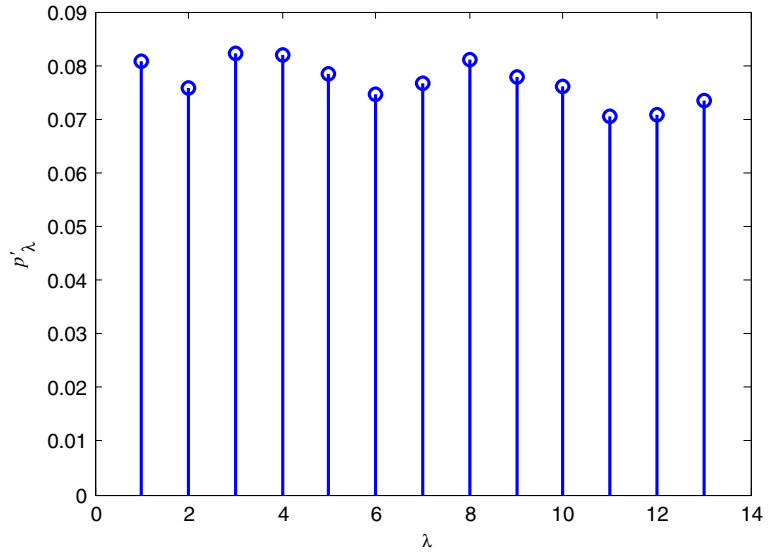

Figure 6 Values of p'_λ under incorrect parameters.

Step 11: Calculate the LCs of the minimal polynomials over GF(2^m) by Equations 47 and 48 for the M packets in W, and get the LC vector as shown in Equation 49.
Step 12: Recognize the generator polynomial follow the steps described in section 4.2.

Finally, we need a detection threshold to reject random data. When the received data stream is not encoded by binary cyclic codes, it can be considered that the data is random for all the coding parameters. The recognizer should give a report to reject the estimated parameters when the parity-check matrix is not likely enough.

We define the mean value of $p'_{j,\lambda}$ for all the blocks in the observed window as follows:

$$\text{mean}\left(p'_{j,\lambda}\right) = \frac{1}{M}\sum_{j=1}^{M} p'_{j,\lambda}, \tag{58}$$

where $p'_{j,\lambda}$ is calculated by Equation 27 according to the recognized parity-check matrix $H\hat{\omega}$, H in Equation 27 is the recognized parity-check matrix $H\hat{\omega}$ and n_r denotes the number of rows of $H\hat{\omega}$. As shown in Figure 4, the distributions of mean $(p'_{j,\lambda})$ for random data and coded data with correctly estimated coding parameters are separated.

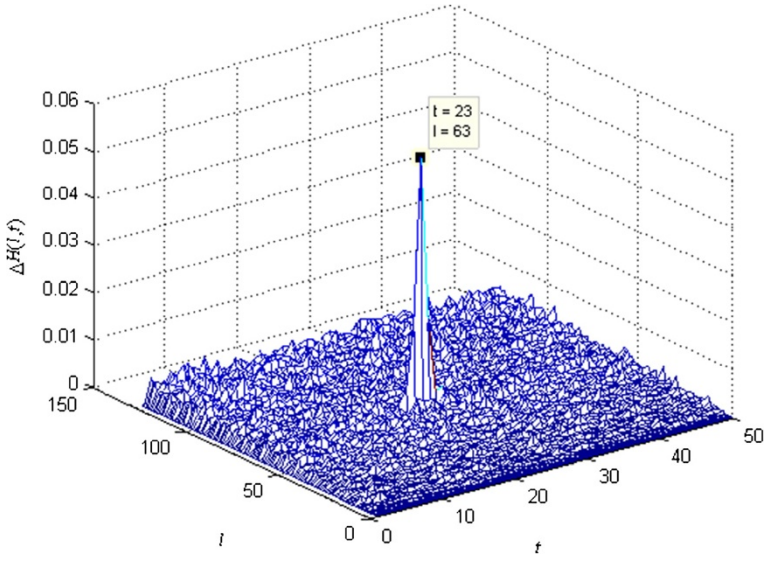

Figure 7 IDEF on different code length and synchronization positions.

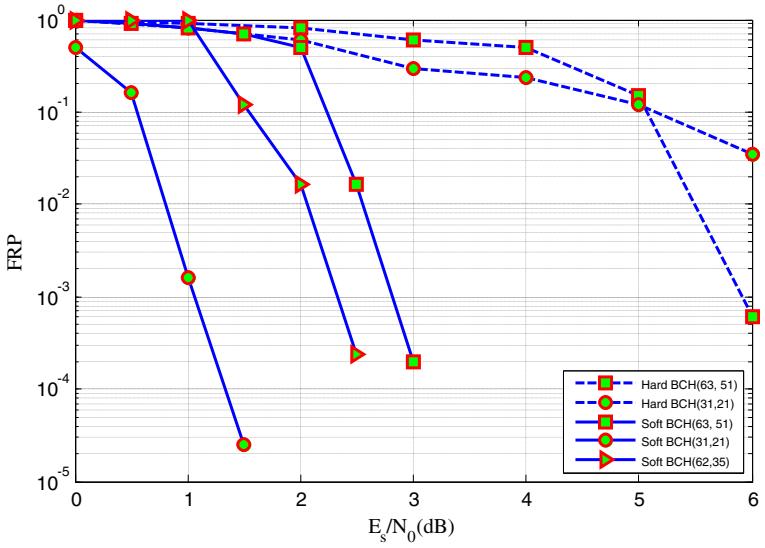

Figure 8 FRP of code length and synchronization position recognization on different SNRs for several binary cyclic codes.

The distances between the two distributions are mainly determined by the noise level, the number of rows in $H\hat{\omega}$, and the number of code blocks in the observed window. Experimentally, we propose the threshold δ to be about 0.6, in order to decide whether the data stream is random or not. After the estimation of the coding parameters, we calculate mean $(p'_{j,\lambda})$ for all complete code blocks in the observed window. If mean $(p'_{j,\lambda})$ is smaller than δ, we propose to reject the recognition result.

6. Simulations

In this section, we show the efficiency of our proposed blind recognition algorithm by simulations. In the simulations, we assume that the searching range of the code length is 7 ~ 128 and the observed window contains N = 3,000 consecutive soft-decision bits from the BPSK demodulator. Meanwhile, we assume the data stream is corrupted by an AWGN on the channel.

When employing the proposed algorithm to recognize the BCH (63, 51) code, the simulation results for code length and synchronization position recognitions are shown in Figures 5, 6, 7 and 8. The SNR is E_s/N_0 = 5 dB and corresponding BER is $10^{-2.19}$. Figure 5 shows the values of p'_{λ} defined in Equation 20 when l = 63 and m = 6, and the block synchronization is achieved. Figure 6 is the case of another l and m. It is shown in

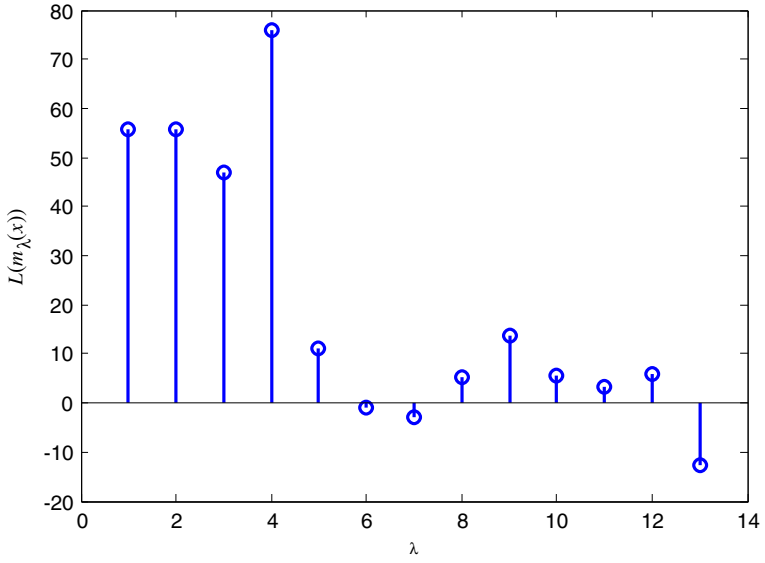

Figure 9 Generator polynomial recognition of cyc (63, 36): original LCs.

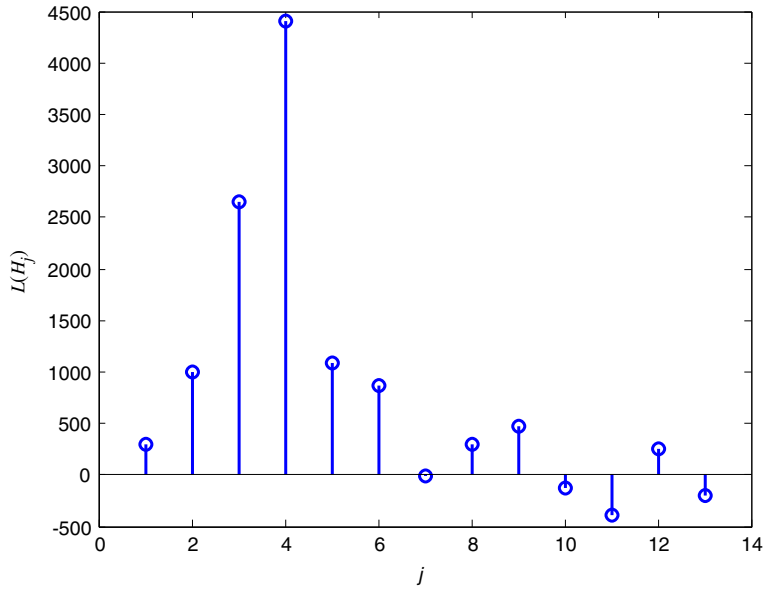

Figure 10 Generator polynomial recognition of cyc (63, 36): sorted LCs.

the two figures that when the code length and synchronization positions are correctly estimated, some minimal polynomials have higher probabilities to be factors of the received codeword polynomials. The obviously larger ones are calculated on the minimal polynomials which are factors of the generator polynomial. If the parameters are not correctly estimated, such feature will not exist. Figure 7 shows the IDEF ΔH for different code length l and synchronization position t, while the first bit of the observed window is the 40th bit of a codeword. When $l = 63$ and $t = 23$, the IDEF is the largest. Thus, we propose $l = 63$ and $t = 23 + lk(k \in \mathbb{Z}^+)$ to be the estimation of the code length and synchronization positions, which are consistent with the simulation settings.

The performance of the algorithm is affected by the channel quality. In Figure 8, we draw the performance of the proposed algorithm when applied to code length recognitions of several different binary cyclic codes. The curves depict the false recognition probabilities (FRP) of the code length and synchronization position estimations on different SNRs. In Figure 8, we also compare the performance of our proposed recognition

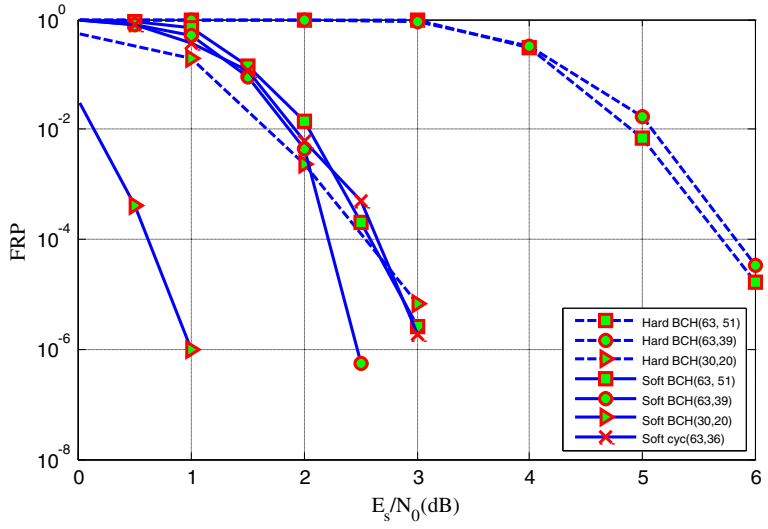

Figure 11 FRP of generator polynomial recognization on different SNRs for several binary cyclic codes.

algorithm with the hard-decision-based RIDERS algorithm proposed in [16-18]. The PFR of our proposed algorithm fall rapidly when SNR increases, and it is much lower than that of the previous algorithms on each single SNR value.

After the code length estimation, the generator polynomial could be recognized by searching for the minimal polynomials which are factors of the generator polynomial according to the steps proposed in section 4.2. We assume that the data stream sent by the transmitter is coded by a cyclic code, the code length and information length of which are 63 and 36, respectively. We call it cyc (63, 36) code in this paper. The generator polynomial of the code is the product of the following minimal polynomials, which includes low-degree minimal polynomials:

$$\begin{cases} m_1(x) = x^6 + x + 1 \\ m_2(x) = x^6 + x^4 + x^2 + x + 1 \\ m_3(x) = x^6 + x^4 + x^2 + x + 1 \\ m_4(x) = x^6 + x^5 + x^2 + x + 1 \\ m_5(x) = x^3 + x^2 + 1 \\ m_{13}(x) = x + 1 \end{cases} \tag{59}$$

The coded data is modulated by BPSK and corrupted by an AWGN with SNR E_s/N_0 = 1.5 dB, and the corresponding hard-decision BER is about 4×10^{-2}. The recognizing procedure is shown in Figures 9, 10 and 11.

There are 13 minimal polynomials over GF(2^6), which are listed below:

$$\begin{cases} m_1(x) = x^6 + x + 1 \\ m_2(x) = x^6 + x^4 + x^2 + x + 1 \\ m_3(x) = x^6 + x^4 + x^2 + x + 1 \\ m_4(x) = x^6 + x^5 + x^2 + x + 1 \\ m_5(x) = x^3 + x^2 + 1 \\ m_6(x) = x^6 + x^5 + x^3 + x^2 + 1 \\ m_7(x) = x^6 + x^4 + x^3 + x^1 + 1 \\ m_8(x) = x^6 + x^5 + x^4 + x^2 + 1 \\ m_9(x) = x^2 + x^1 + 1 \\ m_{10}(x) = x^6 + x^5 + x^4 + x + 1 \\ m_{11}(x) = x^3 + x + 1 \\ m_{12}(x) = x^6 + x^5 + 1 \\ m_{13}(x) = x + 1 \end{cases} \tag{60}$$

Figure 9 shows the original LCs of different minimal polynomials over GF(2^6) to be factors of the codeword polynomials in the observed window. We rank the original LCs from the highest to the lowest, in order to form a new vector L_R and record the index I (defined in Equation 51) as follows:

$$I = [4 \quad 1 \quad 2 \quad 3 \quad 9 \quad 5 \quad 12 \quad 10 \quad 8 \quad 11 \quad 6 \quad 7 \quad 13] \tag{61}$$

Table 1 LCs for H_4 and $H_{4,k}$

H	LCs
H_4	4,406.8
$H_{4,1}$	5,237.4
$H_{4,2}$	468.2
$H_{4,3}$	−389
$H_{4,4}$	5,424.7

Then we let ω increase from 1 to 13, combine the binary minimal parity matrices for the minimal polynomials $m_{I(1)}(x)...m_{I(\omega)}(x)$, in order to form H_ω by Equation 52, and calculate the LCs of $H_\omega \times C_r = 0 (1 \le \omega \le q)$ by Equation 48. The LCs are shown in Figure 10. We can see that the LC of H_4 is the highest. H_4 is obtained by combining the minimal parity-check matrices $Hb_{\min}(m_4(x))$, $Hb_{\min}(m_1(x))$, $Hb_{\min}(m_2(x))$ and $Hb_{\min}(m_3(x))$. Furthermore, we list the low-degree minimal polynomials to check whether they are factors of the generator polynomial. The low-degree minimal polynomials are $m_{L1}(x) = m_5(x)$, $m_{L2}(x) = m_9(x)$, $m_{L3}(x) = m_{11}(x)$ and $m_{L4}(x) = m_{13}(x)$. We record $LC_{max} = LC(H_4) = 4{,}406.8$ and execute the steps 8 ~ 10 described in section 4.2. Finally, we can obtain the values of $LLR(H_{4,k})(1 \le k \le 4)$ in Table 1.

It is obvious that $LLR(H_{4,1}) > 0.9 \times LLR(H_4)$ and $LLR(H_{4,4}) > 0.9 \times LLR(H_{4,1})$. Therefore, $H_{4,4}$ should be considered as the finally recognized parity-check matrix. According to section 4.2, $H_{4,4}$ is obtained by combining the minimal parity-check matrices $Hb_{\min}(m_4(x))$, $Hb_{\min}(m_1(x))$, $Hb_{\min}(m_2(x))$, $Hb_{\min}(m_3(x))$, $Hb_{\min}(m_5(x))$ and $Hb_{\min}(m_{13}(x))$, so we can write the generator polynomial as follows:

$$g(x) = m_1(x)m_2(x)m_3(x)m_4(x)m_5(x)m_{13}(x) \tag{62}$$

The recognition result is accordant with the simulation settings.

Figure 11 shows the performance of the proposed generator polynomial recognition algorithm when applied to several different binary cyclic codes. The curves show the

Table 2 Error rejection rate

E_s/N_0 (dB)	ERP for BCH (63,51)	ERP for BCH (31,21)	ERP for cyc (63,36)	EAP for random data
−1.0	1.00E0	4.06E-1	5.10E-1	<2E-6
−0.5	1.00E0	9.00E-3	4.51E-2	
0.0	9.95E-1	6.67E-6	1.19E-4	
0.5	9.90E-1	<2E-6	<2E-6	
1.0	5.21E-1			
1.5	1.32E-2			
2.0	<2E-6			

FRP on different noise levels. As E_s/N_0 rises, the curves fall rapidly. We also compare our proposed algorithm with the previous hard-decision-based recognition algorithms proposed in [16-18]. It shows that the recognition performance is improved obviously in soft-decision situations.

After the coding parameter recognition, an additional testing program checks whether the data is random. The principle is described in section 4.2. We list the error-rejection-probabilities (ERPs) for some binary cyclic codes and the error-acceptance probabilities (EAP) for random data in Table 2. The ERP level is much lower than the FRP. Especially when the noise level is low enough, the ERPs are nearly zeros. And all the random data is rejected, that is to say, nearly no recognized result on random data is accepted.

7. Conclusion

A blind recognition method for binary cyclic codes for non-cooperative communications and ACM in soft-decision situations is proposed. The code length and synchronization positions are estimated by checking the minimal parity-check matrices. After that, the whole check matrix and generator polynomial are reconstructed by searching which minimal polynomials are factors of the generator polynomial. The recognition method proposed in this paper is based on an earlier published RIDERS algorithm with some significant improvements. By calculating the probability that a minimal polynomial is a factor of the received codewords rather than checking whether an element in the extension field is a root of the codewords, we develop the RIDERS algorithm to soft-decision situations. To calculate the probability that a minimal polynomial is a factor of a received codeword, we adopt some algorithms and ideas introduced in soft-decision-based decoding methods and blind-frame-synchronization approaches for RS and BCH codes in the literatures. Although we have always a loss of performance when these algorithms are applied in cyclic codes while they are particularly well suited for LDPC codes, the algorithm proposed in this paper still has a previously better recognition performance for binary cyclic codes in a soft-decision situation than that in a hard-decision situation. And by the reliability-based adaptive processing, we reduce the influences of the most unreliability decision bits on the calculation of the syndromes, though the parity-check matrices of binary cyclic codes are not sparse. Moreover, the application field of the recognition method is extended to general binary cyclic codes in this paper, including shortened codes. To the best of our knowledge, this paper is the first publication in literature, which introduces an approach for complete-blind recognition of binary cyclic codes in soft-decision situations. Simulations show that our proposed blind recognition algorithm yields obviously better performance than that of the previous ones.

Competing interests
The authors declare that they have no competing interests.

Acknowledgements
This paper was supported by the graduate innovation fund of the National University of Defence Technology. And we wish to thank the anonymous reviewers who helped to improve the quality of the paper.

Received: 19 February 2013 Accepted: 20 August 2013
Published: 30 August 2013

References

1. V Choqueuse, M Marazin, L Collin, KC Yao, G Burel, Blind reconstruction of linear space-time block codes: a likelihood-based approach. IEEE. Trans. Signal. Proc. **58**(3), 1290–1299 (2010)
2. G Burel, R Gautier, Blind estimation of encoder and interleaver characteristics in a non cooperative context, in *Proceedings of IASTED International Conference on Communications, Internet and Information Technology* (Scottsdale, AZ, 2003)
3. M Marazin, R Gautier, G Burel, Algebraic method for blind recovery of punctured convolutional encoders from an erroneous bitstream. IET. Signal. Proc. **6**(2), 122–131 (2012)
4. R Moosavi, EG Larsson, A fast scheme for blind identification of channel codes, in *Proceedings of the 54th GLOBECOM 2011* (Houston, 2011)
5. AJ Goldsmith, SG Chua, Adaptive coded modulation for fading channels. IEEE. Tran. Commun. **46**(5), 595–602 (1998)
6. M Marazin, R Gautier, G Burel, Dual code method for blind identification of convolutional encoder for cognitive radio receiver design, in *Proceedings of IEEE Globecom Workshops* (Honolulu, 2009)
7. F Wang, Z Huang, Y Zhou, A method for blind recognition of convolution code based on Euclidean algorithm, in *Proceedings of IEEE WiCom* (Shanghai, 2007), pp. 21–25
8. J Dignel, J Hagenauer, Parameter estimation of a convolutional encoder from noisy observations, in *Proceedings of IEEE ISIT* (Nice, 2007)
9. M Marazin, R Gautier, G Burel, Blind recovery of the second convolutional encoder of a turbo-code when its systematic outputs are punctured. Mil. Tech. Acad. Rev. **XIX**(2), 213–232 (2009)
10. Z Yongguang, Blind recognition method for the turbo coding parameters. J. Xidian. Univ. **38**(2), 167–172 (2011)
11. M Marazin, R Gautier, G Burel, Blind recovery of k/n rate convolutional encoders in a noisy environment. EURASIP J. Wirel. Commun. Netw. **2011**(168), 1–9 (2011)
12. M Cluzeau, Block code reconstruction using iterative decoding techniques, in *Proceedings of IEEE ISIT* (Seattle, 2006)
13. J Barbier, G Sicot, S Houcke, Algebraic approach for the reconstruction of linear and convolutional error correcting codes, in *Proceedings of World academy of science, engineering and technology* (Venice, Italy, 2006)
14. J Barbier, J Letessier, Forward error correcting codes characterization based on rank properties, in *Proceedings of International Conferences on Wireless Communications* (Nanjing, 2009)
15. Z Junjun, L Yanbin, Blind recognition of low code-rate binary linear block codes. Radio. Eng. **39**(1), 19–22 (2009)
16. W Niancheng, Y Xiaojing, Recognition methods of BCH codes. Elec. Warfare. **2010**(6), 30–34 (2010)
17. Y Xiaojing, W Niancheng, Recognition method of BCH codes on roots information dispersion entropy and roots statistic. J. Detect. Contr. **32**(3), 69–73 (2010)
18. L Xizai, H Zhiping, S Shaojing, Fast recognition method of generator polynomial of BCH codes. J. Xidian. Univ. **38**(6), 187–191 (2011)
19. S Lin, DJ Costello, Costello, Reliability-based soft-decision decoding algorithms for linear block codes, in *Error Control Coding: Fundamentals and Applications*, 2nd edn. (Pearson Pretice Hall, Englewood Cliffs, NJ, 2004), pp. 395–452
20. R Imad, G Sicot, S Houcke, Blind frame synchronization for error correcting codes having a sparse parity check matrix. IEEE. Trans. Comm. **57**(6), 1574–1577 (2009)
21. R Imad, S Houcke, Theoretical analysis of a MAP based blind frame synchronizer. IEEE. Trans. Wireless. Commun. **8**(11), 5472–5476 (2009)
22. R Imad, C Poulliat, S Houcke, G Gadat, Blind frame synchronization of Reed-Solomon codes: non-binary vs. binary approach, in *Proceedings of IEEE SPAWC 2010* (Marrakech, Morocco, 2010)

23. R Imad, S Houcke, C Jego, Blind frame synchronization of product codes based on the adaptation of the parity check matrix, in *Proceedings of IEEE ICC 2009* (Dresden, Germany, 2009)
24. S Lin, DJ Costello, Linear block codes, in *Error Control Coding: Fundamentals and Applications*, 2nd edn. (Pearson Pretice Hall, Englewood Cliffs, NJ, 2004), pp. 66–98
25. S Lin, DJ Costello, Introduction to algebra, in *Error Control Coding: Fundamentals and Applications*, 2nd edn. (Pearson Pretice Hall, Englewood Cliffs, NJ, 2004), pp. 25–65
26. J Jiang, KR Narayanan, Iterative soft-input-soft-output decoding of Reed-Solomon codes by adapting the parity check matrix. IEEE. Trans. Infor. Theory. **52**(8), 3746–3756 (2006)

doi:10.1186/1687-1499-2013-218
Cite this article as: Jing *et al.*: Blind recognition of binary cyclic codes. *EURASIP Journal on Wireless Communications and Networking* 2013 **2013**:218.