Int. J. Communications, Network and System Sciences, 2010, 3, 495-506
doi:10.4236/ijcns.2010.36067 Published Online June 2010 (http://www.SciRP.org/journal/ijcns/).

Performances of Chaos Coded Modulation Schemes Based on Mod-MAP Mapping and High Dimensional LDPC Based Mod-MAP Mapping with Belief Propagation Decoding

Naim Khodor, Jean-Pierre Cances, Vahid Meghdadi, Raymond Quere
University of Limoges, *XLIM-Department*, *Limoges*, *France*
E-mail: {*naim.khodor*, *cances*, *frmeghdadi*, *raymond.quere*}*@ensil.unilim.fr*
Received February 5, 2010; *revised March* 11, 2010; *accepted April* 15, 2010

Abstract

In this paper, we propose to generalize the coding schemes first proposed by Kozic *et al*. to high spectral efficient modulation schemes. We study at first Chaos Coded Modulation based on the use of small dimensional modulo-MAP encoding process and we give a solution to study the distance spectrum of such coding schemes to accurately predict their performances. However, the obtained performances are quite poor. To improve them, we use then a high dimensional modulo-MAP mapping process similar to the low-density generator-matrix codes (LDGM) introduced by Kozic *et al*. The main difference with their work is that we use an encoding and decoding process on GF (2^m) which enables to obtain better performances while preserving a quite simple decoding algorithm when we use the Extended Min-Sum (EMS) algorithm of Declercq & Fossorier.

Keywords: Chaos Coded Modulation, Expectation Maximization, Gaussian or Rayleigh Mixtures, Low-Density Parity-Check (LDPC), Low-Density Generator-Matrix (LDGM), Factor Graph, Extended Min-Sum (EMS)

1. Introduction

Since the pioneering work of Frey in 1993 [1], chaotic communications has been an important topic in digital communications. Due to their extreme sensitivity to initial conditions which, for example, facilitates theoretically the separation of merging paths in a trellis based code, these systems have also been considered as good potential candidates for channel encoding [2-7]. This explains why chaotic modulations and channel encoders derived from chaotic systems have been extensively studied in the open literature. According to us, there are mainly two types of chaos based channel encoders depending on the size of the transmitted alphabet. The first kind of chaos based channel encoders includes non-linear generators which transmit binary messages and benefit from the correlation between successive transmitted bits to obtain some coding gain. Due to the poor spectral efficiency, it is rather easy to optimize this kind of codes to obtain a non-null free distance and to obtain reasonable good performances, *i.e.*, codes that outperform un-coded systems [8-12]. Some authors have even used these binary non-linear constituent encoders to build parallel concatenated schemes just like turbo-codes which perform quite closely to the theoretical bounds provided that the interleave size is big enough [13,14]. The second kind of chaos based channel encoders includes those which transmit a complex quasi-continuous alphabet, *i.e.*, those which are inherently chaotic in all their characteristics. These channel encoders exhibit a high spectral efficiency and can be compared to Trellis Coded Modulation (TCM) schemes. Many works deal with the optimization of such coders and, among them, perhaps the most famous ones were those named Chaos Coded Modulation (CCM) schemes. However, the weakness of such transceiver was their poor BER performance since they did not have even better performances than un-coded systems such as Binary Phase Shift Keying (BPSK) [15-17]. This was particularly the case for the systems which use CSK (Chaos Shift Keying) Modulation [18-20]. Nevertheless, some recent studies have stressed the fact that Chaos Coded Modulation (CCM) systems, working at a joint waveform and coding level, can be efficient in additive white Gaussian noise channels [21-23]. These promising works on the AWGN channel have been recently further extended by Escribano & al in the case of Rayleigh flat fading channels [24].

In this work, we use Chaos Coded Modulation designs of S. Kozic [25,26] and we optimize them using the distance spectrum. We find that the distance spectrum distribution can be good approximated by Rayleigh probability distribution function (pdf). Using this optimization step, we can optimize their structures. Furthermore we show that using a high dimensional modulo-MAP mapping process we are able to considerably improve the performances of this kind of schemes and to obtain performing codes. This principle is related to the former work of Kozic & Hasler in [27]. In their work, low-density generator-matrix (LDGM) codes are used as natural interleave in front of mappings to signal constellation. That is why this kind of code can be assumed as particular kind of BICM. However, the chaotic map is used as joint coding (interleaving) and modulation, and thus, the complete system is a single code. The framework of iterative decoding is based on factor graphs, which is a graphical representation of codes. LDPC and LDGM linear block codes have a very simple graphical representation called Tanner graph. However, for nonlinear codes, the graphical representation is not so simple, and this may be the reason why the large potential of nonlinear codes is not yet exploited. The main difference with their work is that we use an encoding and decoding process on GF (2^m) which enables to obtain better performances while preserving a relative simple decoding algorithm when we use the Extended Min-Sum (EMS) algorithm of Declercq & Fossorier [28]. The contributions of our paper are thus the following ones.

Detailed study of the distance spectra of the chaos based encoders and characterization of their distribution.

New encoding and decoding process based on the use of graph factorization and the use of Belief Propagation (BP) algorithm over high order Galois fields GF (2^m).

The rest of the paper is organized as follows. In Section 2, we give the basic principles for the chaos coded modulation schemes proposed by S. Kozic. We propose to approximate the distance distribution with some usual laws such as the Rayleigh one. In Section 3, we show the high dimensional coding process based on LDPC over GF(2^m); simulation results are provided which demonstrate the outstanding performances of these structures. Concluding remarks are eventually given in Section 4.

2. Chaos Coded Modulation Scheme, Distance Spectrum Study

2.1. Chaotic Coder Structure

We consider the Chaos-Coded modulation scheme of **Figure 1**. This scheme was originally given by S. Kozic in his PhD works [26]. The scheme of **Figure 1**. can be represented by means of a convolutional coder of rate $\eta = 1/(n.(Q+1))$, where at each time step k, one bit b_k enters the coder and a vector of $(Q+1)$ bits v = $[v_Q, v_{Q-1}, \ldots, v_0]^T$ is produced. The signal constellation is realized by a weighted sum of vectors 2^{-i}. $\boldsymbol{A}^{(Q-i+1)}$ *mod* (1) where $\boldsymbol{A}$ is some matrix which optimizes the distance spectrum of the code. This mapping, due to the modulus operation, is a highly non-linear operation and serves as a chaos generator. Henceforth, we have a system which combines a convolutional coder with a multi-dimensional mapping in the same way as Multilevel Trellis Coded Modulation (M-TCM). The corresponding convolutional coder is classically described by:

$$h_i(D) = \frac{v_i(D)}{b(D)} = t_{i,Q} + t_{i,Q-1}.D + \ldots + t_{i,0}.D^Q \quad (1)$$

S. Kozic defines several possible matrices $\boldsymbol{T} = \{t_{i,j}\}$ in his work which give good performances:

$$\boldsymbol{T}_{shift} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \end{bmatrix} \quad \boldsymbol{T}_{e-shift} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & \end{bmatrix}$$

$$\boldsymbol{T}_{tent} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & \end{bmatrix}$$

Concerning, the choice of the matrix $\boldsymbol{A}$, we can write the transmitted vector at the output of the modulator:

$$x_k = \sum_{i=0}^{Q_a} 2^{-(i+1)} A^{Q_a - i} v_i(D) + \sum_{i=Q_a+1}^{Q} 2^{-(i+1)} v_i(D) \bmod(1) \quad (2)$$

Before transmitting $\boldsymbol{x}_k$ on the channel propagation medium, we modulate each of its components in NRZ-BPSK, *i.e.*, : $\boldsymbol{x}_k \rightarrow 2\,\boldsymbol{x}_k - 1$.

Rather than a global optimization algorithm which should look for the convolutional coder together with the mapping process, we choose to fix a convolutional coder structure and then we work on the mapping process by using a particular form of matrix $\boldsymbol{A}$. We found that the choice $\boldsymbol{T}_{i,j} = \boldsymbol{T}_{shift}$ for $i = j$ and $\boldsymbol{T}_{i,j} = \boldsymbol{T}_{tent}$ for $i \neq j$ enables to obtain a large set of performing non-linear mapping with $\boldsymbol{A}$. For example, in the case $n = 2$, using this choice for matrices $\boldsymbol{T}$, we are looking for matrices $\boldsymbol{A}$ with the following structure:

$$\boldsymbol{A} = \begin{pmatrix} 1 & -1 \\ a_{21} & 1 \end{pmatrix}.$$

and we optimize the choice of a_{21} using the distance spectrum. In the case, $n = 3$, we use matrix form:

$$\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 \\ a_{21} & 1 & 1 \\ a_{31} & a_{32} & 1 \end{pmatrix}.$$

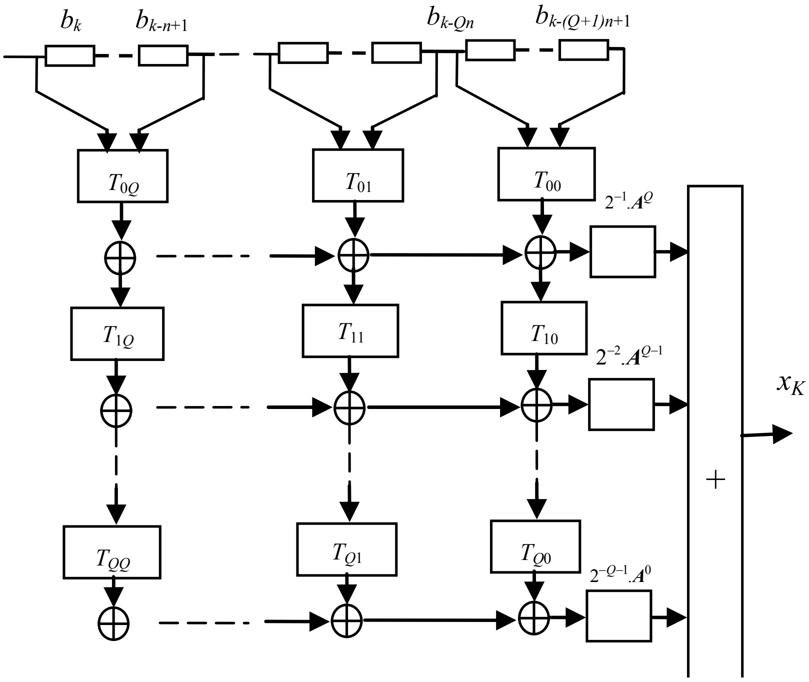

Figure 1. Trellis chaos-coded modulation encoder.

The choice of the remaining parameters $a_{i,j}$ is done using the distance spectrum of the code. The state of the coder is defined by vector $\boldsymbol{S}_k$:

$$\boldsymbol{S}_k = [\mathrm{b}_k,\ldots,b_{k-n},\ldots\mathrm{b}_{k-Qn},\ldots,\mathrm{b}_{k-(Q+1).n+1}]^T \qquad (3)$$

Concerning the choice of Q, it's clear that the Viterbi decoding algorithm is rapidly limited by the complexity in the number of states which is equal to $2^{n.(Q+1)}$. Practically, the number $n(Q + 1)$ should not exceed 12 which correspond to 4096 states. For $n = 2$, this gives a maximum value of Q equal to 5, and for $n = 3$, this gives a maximum value of Q equal to 3. The choice of Q_a, is more complicated and is related to the chaotic behaviour of the coder.

2.2. Spectrum Distance Analysis

In order to optimize the coders, we study their distance spectrum. To do this, we have to determine the trajectories in the trellis which start with a common state $\boldsymbol{S}_i = \boldsymbol{S}_i^*$ and evolve in disjoint paths for (L-1) time steps and then merge again into the same state $\boldsymbol{S}_k = \boldsymbol{S}_k^*$ not necessarily equal to $\boldsymbol{S}_i$. This kind of trajectory in the trellis defines a loop and the loop is characterized by its initial state $\boldsymbol{S}_i$, its final state $\boldsymbol{S}_k$ and its length L. The distance of corresponding codewords belonging to the two competing paths in the loop is:

$$d^2_{L,S_i,S_k} = \sum_{m=1}^{L-1} \left\| \boldsymbol{x}_m - \boldsymbol{x}_m^* \right\|^2 \qquad (4)$$

The problem of the computation of (4) is that, unlike linear codes when we can choose a reference path equal to a all zero sequence, due to the non-linear mapping, we have to test all the possible transmitted sequence for a given loop length together with all the possible starting states. Hence, the distance spectrum computation problem is of non polynomial complexity and in straightforward manner requires the inspection of all possible initial conditions and all possible controlled trajectories. For example, there are $2^{n.(Q+1)}.2^{nL}$ different controlled trajectories of length L. In order to compute the distance spectrum with a reasonable complexity while keeping a sufficient accuracy, we form all the possible pair of sequences starting from a given state and both converging towards an other state after L steps with L belonging to the interval $[Qn + 1, n.(Q + m)]$, *i.e.* the length of the loop varies from Qn+1 (the constraint length of the code plus one) to to $n.(Q + m)$ (we limit practically the search to $m = 2$ or 3 in our case due to the computation burden). We have partitioned the distance spectrum into subsets by distinguishing error events which entail one error bit, error events which entail two error bits, error events which entail three error bits and so on. In practice, we limit our search to error events which entail five maximum error bits since simulation results evidenced that it was sufficient to obtain accurate upper bounds for the BER.

We obtain for example with matrices: $\boldsymbol{T}_{i,j} = \boldsymbol{T}_{shift}$ for $i = j$ and $\boldsymbol{T}_{i,j} = \boldsymbol{T}_{tent}$ (*i.e.* $n = 2$) for $i \neq j$ and $a_{21} = 8$, $Q = Q_a = 3$, the distance spectrum illustrated on **Figure 2**.

In fact, we found that, in a majority of cases, the shape of the distance spectrum is close to a Rayleigh distribution with the following probability density function:

$$f_C(x) = \frac{(x-\mu_j)}{\sigma_j^2}.e^{-(x-\mu_j)^2/2.\sigma_j^2} \quad x \ge \mu_j$$
$$f_C(x) = 0 \quad x < \mu_j \tag{5}$$

For example, with the distance spectrum plotted on **Figure 2**, we calculate parameters μ_j and σ_j^2 to obtain the best fitting between the pdf of the distance spectrum and $f_c(x,m_j,\sigma_j^2)$ we obtain with classical MMSE technique: $\mu_j \cong \sigma_j^2 \cong 6.7$. This corresponds to a minimum free distance of the coder equal to $d_{free} \cong 6.7$. We have developed an original EM (Expectation-Maximization) algorithm to obtain the approximated Rayleigh distribution of the distance spectrum as a mixture of Rayleigh laws. The mixture of Rayleigh laws can be written as:

$$\begin{aligned} f_C(x) &= \sum_{n=1}^{J} \pi_n.\frac{(x-\mu_n)}{\sigma_n^2}.e^{-(x-\mu_n)^2/2.\sigma_n^2} \\ &= \sum_{n=1}^{J} \pi_n.\mathcal{R}(\mu_n,\sigma_n^2) \end{aligned} \tag{6}$$

where $\mathcal{R}(\mu_n,\sigma_n^2)$ represents a Rayleigh law of parameters: μ_n and σ_n^2. The Maximum Likelihood (ML) research algorithm to find: π_n, μ_n, σ_n^2 can be summarized as:

$$\begin{aligned} \hat{\boldsymbol{\theta}} &= \arg \max_{\boldsymbol{\theta}:\sum_{j=1}^{J}\pi_j=1} \log p_{\boldsymbol{\theta}}(\Xi) \\ &= \arg \max_{\boldsymbol{\theta}:\sum_{j=1}^{J}\pi_j=1} \sum_{i=1}^{n} \log \sum_{j=1}^{J} \pi_j.\psi(\xi_i;\mu_j,\sigma_j^2) \end{aligned} \tag{7}$$

where $\psi\,(x,\mu,\sigma^2)$ denotes the value of a Rayleigh law of parameters μ,σ^2 at x. For a fixed number of mixtures J, based on the observations: $\Xi \equiv \{\xi, i = 1,\ldots,n\}$, the parameters $\boldsymbol{\theta} \equiv \{\pi_j, m_j, \sigma_j, j=1,\ldots,J\}$ can be estimated using the EM (Expectation Maximization) algorithm. The algorithm proceeds in two steps:

E-step: Compute

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(i)}) = E_{\boldsymbol{\theta}^{(i)}}\{\log p_{\boldsymbol{\theta}}(\boldsymbol{X})|\,\Xi\} \tag{8}$$

M-step: solve

$$\boldsymbol{\theta}^{(i+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(i)}) \tag{9}$$

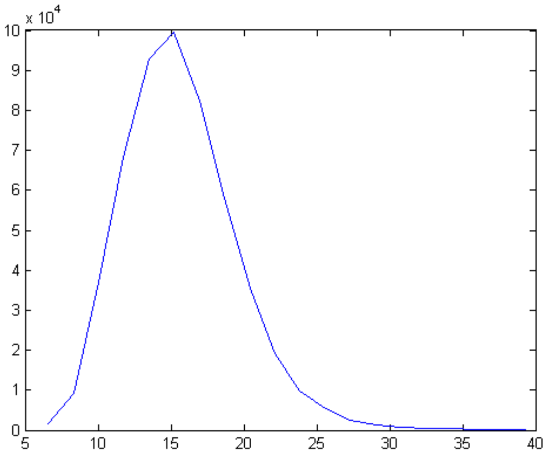

Figure 2. Distance spectrum of the chaos coded modulation.

Define the following hidden data $\boldsymbol{Z} = \{z_i, i = 1,\ldots, n\}$ where z_i is a J-dimensional indicator vectoring such that:

$$z_{i,j} = \begin{cases} 1, \, if \; \xi_i \cong R\,(m_j,\sigma_j^2) \\ 0\,, \, otherwise \end{cases} \tag{10}$$

The complete data is then $\boldsymbol{X} \cong (\Xi,\boldsymbol{Z})$, we have :

$$p_{\boldsymbol{\theta}}(\Xi,\boldsymbol{Z}) = \prod_{i=1}^{n}\prod_{j=1}^{J}\left[\pi_j.R(\xi_i;m_j,\sigma_j^2\right]^{z_{i,j}} \tag{11}$$

The log-likelihood function of the complete data is then given by:

$$\log p_\theta\left(\Xi,Z\right) = \sum_{i=1}^{n}\sum_{j=1}^{J} z_{i,j}.\log\pi_j + \sum_{i=1}^{n}\sum_{j=1}^{J} z_{i,j}.\left[\log\left(\xi_i-\mu_j\right)-\log\left(\sigma_j^2\right)-\frac{\left(\xi_i-\mu_j\right)^2}{2\sigma_j^2}\right]+C \tag{12}$$

where C is a constant. The E-step can then be calculated as follows:

$$Q(\boldsymbol{\theta}|\,\boldsymbol{\theta}') = E_{\boldsymbol{\theta}}^{'}\{\log p_{\boldsymbol{\theta}}(\Xi,\boldsymbol{Z})|\,\Xi\}$$

$$Q(\boldsymbol{\theta}|\,\boldsymbol{\theta}') = \sum_{i=1}^{n}\sum_{j=1}^{J}\hat{z}_{i,j}.\left[\log\pi_j+\log\left(\xi_i-\mu_j\right)-2\log\left(\sigma_j\right)-\frac{\left(\xi_i-\mu_j\right)^2}{2\sigma_j^2}\right]+C$$

We have:

$$\begin{aligned} \hat{z}_{i,j} &= E_{\boldsymbol{\theta}}^{'}\left\{z_{i,j}\,\middle|\;\Xi,\boldsymbol{\theta}^{'}\right\} = P_{\boldsymbol{\theta}}^{'}\left\{z_{i,j}=1\middle|\xi_i\right\} \\ &= \frac{\psi(\xi_i;m_j^{'},\sigma_j^{'2}).\pi_j^{'}}{\sum_{l=1}^{J}\psi(\xi_i;m_l^{'},\sigma_l^{'2}).\pi_l^{'}} \end{aligned} \tag{13}$$

The M-step is calculated as follows. To obtain $\{\pi_j\}$, we have:

$$\frac{\partial Q(\boldsymbol{\theta},\boldsymbol{\theta}^{'})}{\partial\pi_j} = 0 \;=>\; \pi_j = \frac{1}{n}.\sum_{i=1}^{n}\hat{z}_{i,j}, \quad j=1,\ldots,J \tag{14}$$

To obtain $\{\mu_j\}$, we have:

$$\frac{\partial Q(\boldsymbol{\theta},\boldsymbol{\theta}^{'})}{\partial m_j} = 0 \;=>$$

$$\sum_{i=1}^{n}\hat{z}_{i,j}.[-\frac{1}{\xi_i-m_j}+\frac{(\xi_i-m_j)}{\sigma_j^2}] = 0, \quad j=1,\ldots,J$$

$$\sum_{i=1}^{n}\hat{z}_{i,j}.[\frac{-\sigma_j^2+(\xi_i-m_j)^2}{\sigma_j^2.(\xi_i-m_j)^2}] = 0, \quad j=1,\ldots,J$$

$$\sum_{i=1}^{n}\hat{z}_{i,j}.\sigma_j^2=\sum_{i=1}^{n}\hat{z}_{i,j}.(\xi_i^2-2.m_j.\xi_i+m_j^2),\quad j=1,\dots,J$$

$$\sum_{i=1}^{n}\hat{z}_{i,j}.\xi_i^2-2.m_j.\sum_{i=1}^{n}\hat{z}_{i,j}.\xi_i+ \qquad (15)$$

$$m_j^2.\sum_{i=1}^{n}\hat{z}_{i,j}=\sum_{i=1}^{n}\hat{z}_{i,j}.\sigma_j^2,\quad j=1,\dots,J$$

To obtain $\{\sigma_j\}$ we have the set of equations:

$$\frac{\partial Q(\boldsymbol{\theta},\boldsymbol{\theta}')}{\partial\sigma_j}=0$$

$$\sum_{i=1}^{n}\hat{z}_{i,j}.[-\frac{2}{\sigma_j}+\frac{(\xi_i-m_j)^2}{\sigma_j^3}]=0,\quad j=1,\dots,J$$

$$=>2.\sigma_j^2.\sum_{i=1}^{n}\hat{z}_{i,j}=\sum_{i=1}^{n}\hat{z}_{i,j}.(\xi_i-m_j)^2,\quad j=1,\dots,J \qquad (16)$$

$$\sigma_j^2=\frac{\sum_{i=1}^{n}\hat{z}_{i,j}.(\xi_i-m_j)^2}{2.\sum_{i=1}^{n}\hat{z}_{i,j}},\quad j=1,\dots,J \qquad (17)$$

The set of Equations (16) and (17) is a set of coupled non-linear equations and we use the optimization toolbox with the function *fsolve* to solve (16-17) at each maximization step.

2.3. Performances over AWGN Channels

To end this part, we give some BER results on AWGN channels, using the optimization obtained by the distance spectrum computation to find good modulation parameters. Due to a lack of place we only give simulation results. For $n = 2$, we obtain the following result on **Figure 3**.

The chaotic coder outperforms un-coded BPSK at high SNR's due to good asymptotic properties with a moderate high free distance. The weakness of this kind of code is their poor coding rate. There are several solutions to improve this. The first is to make input bits enter the coder by groups of k bits. In this case, the coding rate becomes equal to: $k/n.(Q + 1)$.

However, this considerably reduces the correlation degree between consecutive states and renders the trellis non-binary. We found that the penalty encountered by this method too much important (using $k = 2$ results in 4 dB losses compared to $k = 1$) so we prefer using puncturing to increase the coding rate of our proposed coders. We added the case of punctured codes on **Figure 4** with the best puncturing patterns we found for rate 8/7 and 4/3.

The conclusions are nearly the same, considering the case $n = 3$ as it is illustrated on **Figure 4**.

With a free distance equal to 13 for the optimized code with rate 1/9, the punctured codes (9/7) are able to out perform un-coded BPSK at high SNR's. To complete this overview of BER performances over AWGN channels, it is important to say that using the approximate pdf's of the distance spectrum, we are able to accurately predict the BER at high's SNR's. To complete the results, we give on **Figure 12** the best performances we found with $n = 3$, $Q = 3$ (*i.e.* the number of states is 4096).

In fact, as it is expected, increasing the quantization level for a given dimensionality n, entails some losses. Compared to **Figure 4**, the loss in terms of SNR for a BER of 10^{-4}, 10^{-5} is approximately 1 dB on **Figure 5** and punctured codes are unable to outperform un-coded BPSK.

It is clear that the obtained performances remain poor.

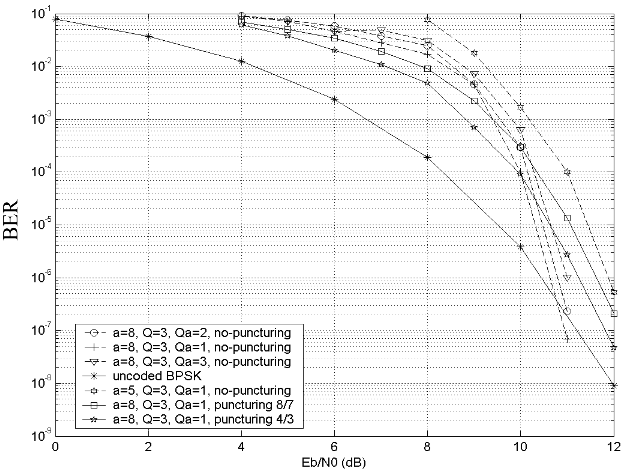

Figure 3. Performances of Trellis Chaos-Coded Modulation over AWGN channels for n = 2, Q = 3.

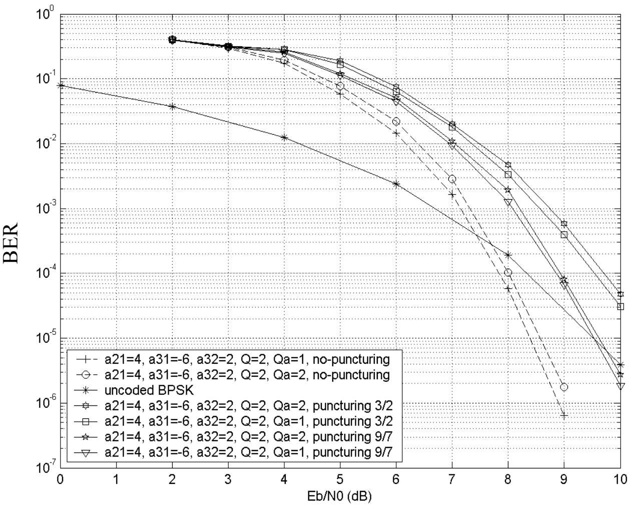

Figure 4. Performances of Trellis Chaos-Coded Modulation over AWGN channels for n = 3, Q = 2.

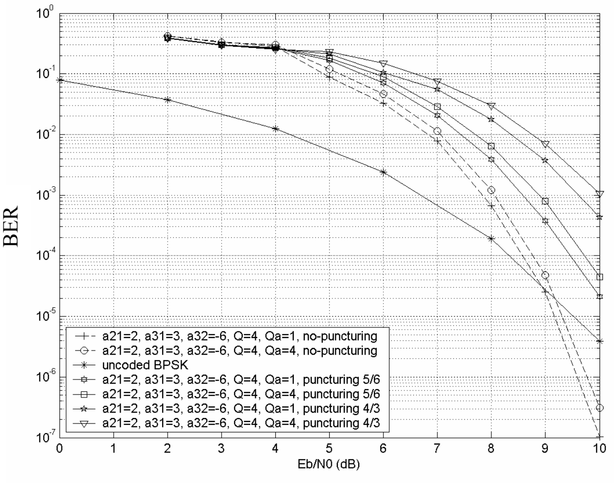

Figure 5. Performances of optimized Trellis Chaos-Coded Modulation over AWGN channels for *n* = 3, *Q* = 3 (4096 states).

To increase them, we propose to generalize the non-linear output mapping to matrices A of high-dimension.

3. High Dimensional LDPC Based Mod-MAP Mapping with B.P Decoding

3.1. The Encoding Process

The generalized mod-MAP function is written as:

$$\boldsymbol{x}_{k+1} = 2.\boldsymbol{A}.\boldsymbol{x}_k \bmod q \qquad (18)$$

where $\boldsymbol{x}_k$ is the input vector of size n and $\boldsymbol{A}$ is a $n \times n$ matrix with elements belonging to alphabet: A = (0, 1,…, q-1) with $q = 2^m$ since we take here: $q = 2^m$ for the considered Galois-field $GF(q)$. The encoding scheme is drawn on **Figure 6**. The binary streams $\boldsymbol{b} = (\boldsymbol{b}_1, \boldsymbol{b}_2,\ldots, \boldsymbol{b}_k,\ldots)$ are grouped into vector vector $\boldsymbol{b}_{k+i-Q}$ of size: $n_{\cdot m} = n.m$ with m denoting the spectral efficiency we want to use in the encoding-decoding process. Hence, we can write: $\boldsymbol{b}_{k+i-Q} = (\boldsymbol{b}_{k+i-Q}{}^{(1)}, \boldsymbol{b}_{k+i-Q}{}^{(2)},\ldots, \boldsymbol{b}_{k+i-Q}{}^{(nm-1)}, \boldsymbol{b}_{k+i-Q}{}^{(nm)})^T$. A binary/$q$-ary converter is then used to obtain vectors $\boldsymbol{d}_{k+i-Q} = (\boldsymbol{d}_{k+i-Q}{}^{(1)}, \boldsymbol{d}_{k+i-Q}{}^{(2)},\ldots, \boldsymbol{d}_{k+i-Q}{}^{(\text{n-1})}, \boldsymbol{d}_{k+i-Q}{}^{(\text{n})})^T$ of q-ary symbols: $\boldsymbol{d}_i^{(p)}$, $p = 1,2,\ldots,n$ belonging to the alphabet: A = (0,1,…,q-1). Each obtained vector $\boldsymbol{d}_{k+i-Q}$ is then multiplied by a sparse low-density based matrix $\boldsymbol{A}^{Q-I}$ and weighted by a factor $2^{-(i+1)}$ then, the new resulting encoding vectors are added to obtain the vector:

$$z_k = \sum_{i=0}^{Q} 2^{-(i+1)}.\boldsymbol{A}^{Q-i}.\boldsymbol{d}_{k+i-Q} \bmod(q)\,.$$

Finally, to obtain a chaotic trajectory we add the vector: $\mathbf{2}^{-(\boldsymbol{Q}+1)}.(\boldsymbol{A}\text{-}\boldsymbol{I}).\boldsymbol{e}$ with: $\boldsymbol{e}$ **[1, 1,…, 1]**^{T}. This yields to the following equation:

$$\boldsymbol{x}_k = \sum_{i=0}^{Q} 2^{-(i+1)}.\boldsymbol{A}^{Q\text{-}i}.\boldsymbol{d}_{k+i-Q} + \frac{2^{-Q}}{2}.\boldsymbol{p} \bmod q \qquad (19)$$

With: $\boldsymbol{p}\mathbf{=2}^{-\boldsymbol{Q}}\mathbf{.(}\boldsymbol{A}\text{-}\boldsymbol{I}\mathbf{).}\boldsymbol{e}\mathbf{=}\boldsymbol{K}\mathbf{.(}\boldsymbol{A}\text{-}\boldsymbol{I}\mathbf{).}\boldsymbol{e}$.

The coding sequence at the output of the modulator is then equal to: $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2,\ldots, \boldsymbol{x}_k,\ldots)$.The relation (19) is called the high-dimensional expansion associated with the chaotic system (18). Another way to represent the encoding rule is to use the following equation:

$$\boldsymbol{x}_{k+1} = 2.\boldsymbol{A}.\boldsymbol{x}_k \bmod q + 2^{-Q}.(\boldsymbol{d}_{k+1} - 1/2.\boldsymbol{e}) \bmod q \qquad (20)$$

The relation (20) is simpler than (19) to better understand the encoding algorithm; however Equation (19) is more suitable for the factorization of the factor graph. The rule (20) represents a dynamical system controlled with stochastic perturbations of small amplitudes $2^{-\boldsymbol{Q}}$ which constitute the input signal. It is obvious that as: $Q \rightarrow +\infty$, the small amplitudes vanish and the output signal vector becomes the chaotic state.

It is important to note that the decoding of a LDPC over $GF(q)$ implies that we use a systematic form for the encoding process. It's the case here since $\boldsymbol{b} = (\boldsymbol{b}_1, \boldsymbol{b}_2,\ldots, \boldsymbol{b}_k,\ldots)$ is the systematic part and $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2,\ldots, \boldsymbol{x}_k,\ldots)$ is the redundancy check part. We have of course a overall coding rate of 0.5 because the systematic part and the redundancy check part have the same dimension.

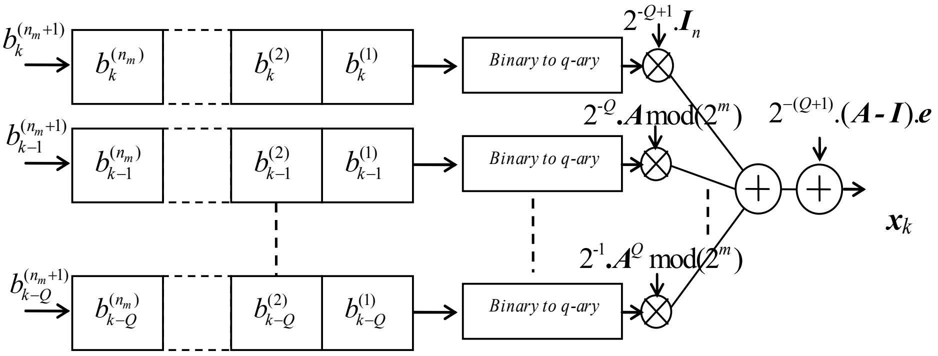

Figure 6. Coding Scheme with high dimensional LDPC based Mod-MAP mapping.

3.2. Factor Graphs

To explain how the factorization can be efficiently implemented it is convenient to represent a function with a factor graph. Once a factor graph has been found it is straightforward to use the BP (Belief Propagation) algorithm to determine the marginal of a multivariate function. For linear block codes the factor graph of the code becomes a Tanner graph. Of course, in our case due to the use of a non-linear encoding process, finding this graph is a much more complicated task. In fact as Kozic demonstrated in [27], there are mainly two possible solutions to obtain it here. The first one consists in using the party-check equation given by Equation (20). The second one is the consequence of the high-dimensional expansion associated with Equation (19). The first solution is not appropriate since it would imply to obtain information about $\boldsymbol{b}_k$ from the soft information about the states $\boldsymbol{x}_k$ which constitute the graph of variable and check nodes and $\boldsymbol{d}_k$ is multiplied by a small value: 2^{-Q}. Hence the reliability about information concerning $\boldsymbol{b}_k$ would be small in this case. Hence, the second solution is the only tractable one. However, it is important to avoid the use of successive power of $\boldsymbol{A}$ in the graph factorization. This is due to the fact that short cycles of length four appear when we use for example $\boldsymbol{A}^2$ in a factor graph even if $\boldsymbol{A}$ does not exhibit short length cycles.

The graph factorization may be expressed in the following way: it comprises mainly three steps. The first one is related directly to the scheme of **Figure 6** and concerns the computation of $\boldsymbol{x}_k$ given $\boldsymbol{d}_{k+i-Q}$ it will be named high-order expansion graph. The second one concerns the LDPC code contained in each matrix $\boldsymbol{A}$, it will be named *GF*(*q*) LDPC graph and finally the third one concerns the way the input bits slide to constitute a new vector to be encoded. This mechanism is related to the convolutional encoder behaviour and will be named convolutional graph.

The high-order expansion graph constitutes the main original part and it can be obtained as follows. We consider at first an indicator function of high dimensional *q*-ary expansion:

$$g(\boldsymbol{x}_k, \boldsymbol{d}_k, \ldots, \boldsymbol{d}_{k-Q}) = \begin{cases} 1, \text{ if } \boldsymbol{x}_k = \sum_{i=0}^{Q} 2^{-(i+1)}.\boldsymbol{A}^{Q-i}.\boldsymbol{d}_{k+i-Q} + \frac{q}{2}.\boldsymbol{p} \bmod q \\ 0, \text{ otherwise} \end{cases} \quad (21)$$

We can use then additional variables $\boldsymbol{\mu}_{i,j}$ defined as:

$$\boldsymbol{x}_k = \sum_{i=0}^{Q} \boldsymbol{\mu}_{i,Q-i} + 2^{-(Q+1)}.\boldsymbol{p} \bmod q$$

Of course, we have the relationship: $\boldsymbol{\mu}_{i,j+1} = \boldsymbol{A}.\ \boldsymbol{\mu}_{i,j}.\text{mod}\ q$ with: $\boldsymbol{\mu}_{i,0} = 2^{-(i+1)}\boldsymbol{d}_{k+i-Q}$. With these variables, function g becomes a function only of variables: $\boldsymbol{\mu}_{i,j}$. To keep on factorizing g we introduce functions $\boldsymbol{g}_{i,j+1}$ defined as :

$$g_{i,j+1}(\boldsymbol{\mu}_{i,j+1}, \boldsymbol{\mu}_{i,j}) = \begin{cases} 1, \text{ if } \boldsymbol{\mu}_{i,j+1} = \boldsymbol{A}.\boldsymbol{\mu}_{i,j} \bmod q \\ 0, \text{ otherwise} \end{cases} \quad (22)$$

and g_0 :

$$g_0(\boldsymbol{x}_k, \boldsymbol{\mu}_{0,Q}, \boldsymbol{\mu}_{1,Q-1}, \ldots) = \begin{cases} 1, \text{ if } \boldsymbol{x}_k = \sum_{i=0}^{Q} \boldsymbol{\mu}_{i,Q-i} + 2^{-(Q+1)}.\boldsymbol{p} \bmod q \\ 0, \text{ otherwise} \end{cases}$$

The corresponding factorization of g is given by:

$$g(.) = g_0 \cdot \prod_{i=0}^{Q} \prod_{j=0}^{Q-i} g_{ij}(.) \quad (23)$$

The factorization is drawn below on **Figure 7**.

It is possible to further factorize the class of functions: $g_{\mathrm{i},j+1}$. The variables at the left side of Equation (22) will be named the checks and the variables on the right side will be considered as the noisy symbols. We define similarly as in the case of LDPC codes: $\mathcal{N}(l) = \{m{:}a_{lm} \neq 0\}$ the set of noisy symbols that participate in the check l. In the same way, we define: $\mathcal{M}(m) = \{l{:}a_{lm} \neq 0\}$ the set of checks that depend on the noisy symbol m. In this case (22) can be written as:

$$\boldsymbol{\mu}_{i,j+1}^{(l)} = \sum_{m \in \mathcal{N}(l)} a_{lm}.\boldsymbol{\mu}_{i,j}^{(m)} \bmod q \quad m,l \in [1,n] \times [1,n] \quad (24)$$

Let:

$$g_{i,j+1}^{(l)} = \begin{cases} 1, \text{ if } \boldsymbol{\mu}_{i,j+1}^{(l)} = \sum_{m \in \mathcal{N}(l)} a_{lm}.\boldsymbol{\mu}_{i,j}^{(m)} \bmod q \\ 0, \text{ otherwise} \end{cases} \quad (25)$$

The symbols on **Figure 7** correspond either to variable nodes (circle on the figure) or check nodes (square on the figure). The symbolise decoding of the complete chaotic trajectory is given by:

$$\begin{aligned} &\arg_{d^{(.)}_{k+i-Q \in \{0,1\}}} \max \sum_{\approx d^{(.)}_{k+i-Q}} p(\boldsymbol{x}_0) \prod_{j=1}^{M+Q} p(\boldsymbol{y}_j | \boldsymbol{x}_j) \\ &\times g(\boldsymbol{x}_j, \boldsymbol{d}_j, \ldots, \boldsymbol{d}_{j-Q}) \\ &\times p(\boldsymbol{d}_j, \ldots, \boldsymbol{d}_{j-Q}) \times p(\boldsymbol{x}_{j+1} | \boldsymbol{x}_j, \boldsymbol{d}_{j+1}). \end{aligned} \quad (26)$$

The quantity ≈ $\boldsymbol{d}_{k+1-Q}$ means summation over all components except: $\boldsymbol{d}_{k+1-Q}$. Furthermore, the graph of matrix $\boldsymbol{A}$ is classically those of a LDPC code over *GF*(*q*) and is drawn on **Figure 8**.

The shift register and the binary *q*-ary conversion set operation, which represent transition state from time j to time $j + 1$ can be given by the indicator function: $g_c = p(\boldsymbol{x}_{j+1}|\boldsymbol{x}_j, \boldsymbol{d}_{j+1})$. The state at time $j + 1$ depends on the symbol sequence: $\boldsymbol{d}_{j+1}, \ldots, \boldsymbol{d}_{j+1-Q}$, and it can be computed using likelihoods of symbols $\boldsymbol{d}_j, \ldots, \boldsymbol{d}_{j+1-Q}$ and additional likely-

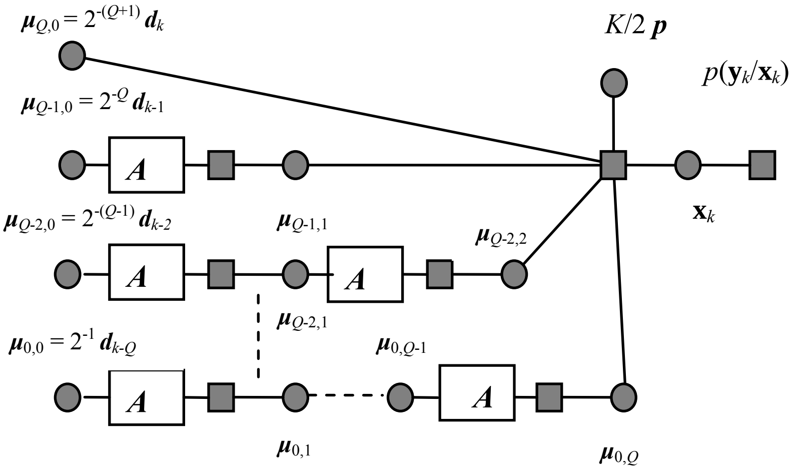
$\mu_{Q,0} = 2^{-(Q+1)} d_k$
$K/2\ p$
$p(y_k/x_k)$
$\mu_{Q-1,0} = 2^{-Q} d_{k-1}$
A
x_k
$\mu_{Q-2,0} = 2^{-(Q-1)} d_{k-2}$
$\mu_{Q-1,1}$
$\mu_{Q-2,2}$
$\mu_{Q-2,1}$
$\mu_{0,0} = 2^{-1} d_{k-Q}$
$\mu_{0,Q-1}$
$\mu_{0,1}$
$\mu_{0,Q}$

Figure 7. high-order expansion factor graph.

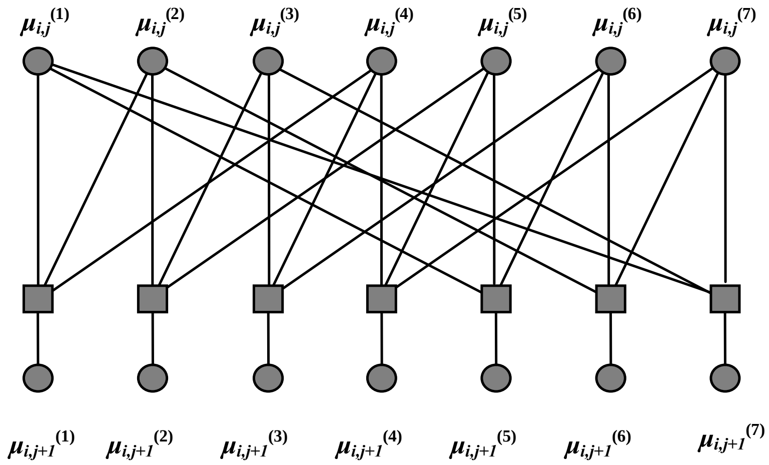

Figure 8. Factor graph for matrix *A*.

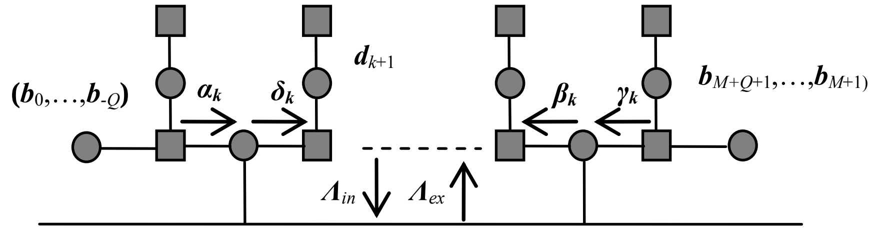

Figure 9. Factor graph of the complete chaotic code.

about symbol: $\boldsymbol{d}_{j+1}$. Using together factorization g_c and factorization of the high-order expansion of **Figure 7**, the decoding problem of (26) can be presented with the factor graph of **Figure 9**. This graph takes into account the shift register process which includes new incoming bits into the encoding process and consists of two recursions: forward and backward recursions as in the well known BCJR algorithm.

The parameters α_k, β_k, γ_k and δ_k are defined in the same way as in [27] except that we work at symbol level for the computation of α_k, β_k, γ_k and δ_k.

Note that since the main difference with the scheme proposed by Kozic, *et al.* in [27] consists in the use of a non-binary encoding scheme, we have to transform a priori probabilities on bits to a priori probabilities in symbols. This is done using the formula:

$$P[\boldsymbol{d}_{k+i-Q} = (b^{(1)}_{k+i-Q}, b^{(2)}_{k+i-Q}, \ldots, b^{m}_{k+i-Q})^T] = \prod_{j=1}^{m} \frac{e^{b^{(j)}_{k+i-Q}\cdot\tilde{\Lambda}_e(b^{(j)}_{k+i-Q})}}{1+e^{b^{(j)}_{k+i-Q}\cdot\tilde{\Lambda}_e(b^{(j)}_{k+i-Q})}} \qquad (27)$$

where $\tilde{\Lambda}_e(b_{k+i-Q}^{(j)})$ designs the log-likelihood ratio corresponding to bit: $b_{k+i-Q}{}^{(j)}$. For the complementary problem, *i.e.* when we have to express the log-likelihood ratio of bit $b_{k+i-Q}{}^{(j)}$ from the log-likelihood ratios of corresponding symbols, we have:

$$\Lambda_e(b_{k+i-Q}^{(j-1).m+k}) = \log \frac{\sum\limits_{d_{k+i-Q}^{(j)}:b_{k+i-Q}^{(j-1).m+k}=1} e^{\Lambda_e(d_{k+i-Q}^{(j)})}}{\sum\limits_{d_{k+i-Q}^{(j)}:b_{k+i-Q}^{(j-1).m+k}=0} e^{\Lambda_e(d_{k+i-Q}^{(j)})}} \quad (28)$$

where, obviously, $\Lambda_e(d_{k+i-Q}^{(j)})$ corresponds to the log-likelihood ratio of symbol:

$$d_{k+i-Q}^{(j)} = [b_{k+i-Q}^{(j-1).m+1}, b_{k+i-Q}^{(j-1).m+2}, \ldots, b_{k+i-Q}^{(j-1).m+m}] .$$

3.3. Iterative Decoding

The main steps of the iterative decoding are the same as those of [27] except for the use of a non-binary generator matrix ***A***. The decoding of LDPC codes over *GF*(q) has been an extensive research topic recently. Among all the decoding algorithms, we choose the Extended Min-Sum (EMS) Algorithm in the log-domain proposed by Declercq and Fossorier [28] since it exhibits a good trade-off between performance and complexity. To explain the main principles we use the following notations. A parity node in a LDPC code over *GF*(q) with $q = 2^m$ represents the following parity equation:

$$\sum_{k=1}^{d_c} h_k(x).i_k(x) = 0 \mod m(x) \quad (29)$$

where $m(x)$ in the modulo operator is a degree m -1 primitive polynomial of *GF*(q). Equation (29) expresses that the variable nodes needed to perform the BP algorithm on a parity node are the codeword symbols multiplied by non-zeros values of the parity matrix ***H***. The corresponding transformation of the graph is performed by adding variable nodes corresponding to the multiplication of the codeword symbols $i_k(x)$ by their associated nonzero ***H*** values and is illustrated on **Figure 10**.

The function node that connects the two variable nodes $i_k(x)$ and $h_k(x)$. $i_k(x)$ performs a permutation of the message values. The permutation that is used to update the message corresponds to the multiplication of the tensor indices by $h_k(x)$.from node $i_k(x)$ to node $h_k(x)$. $i_k(x)$ and to the division of the indices by $h_k(x)$ the other way. With this transformation of the factor graph, the parity node update is indeed a convolution of all incoming messages as in the binary case.

To express the EMS algorithm, we use the following notations for the messages in the graph. Let $\{V_{pv}\}$ v = 1,…,d_v be the set of messages entering a variable node of degree d_v, and $\{U_{vp}\}$v = 1,…,d_v be the output messages for this variable node. The index '*p*.v' indicates that the message comes from a permutation node to a variable node, and 'v.*p*' is for the other direction. We define similarly the messages $\{U_{pc}\}$ c = 1,…,d_c (resp. $\{V_{pc}\}$ c = 1,…,d_c) at the input (resp. output) of a degree d_c check node. The EMS algorithm works in the log-domain and uses reduced configuration sets to simplify the computational task. We define then:

$$\bar{U}[i_1 \ldots i_m] = \log\left(\frac{U[i_1,\ldots,i_m]}{U[0,\ldots,0]}\right) \quad \forall (i_1,\ldots,i_m) \in \{0,1\}^m \quad (30)$$

As the log-density-ratio (LDR) representations of the messages. In the considered q-ary case, the message is composed of q -1 nonzero LDR values. The purpose of the EMS algorithm is to simplify the parity check node update by selecting only the most probable configuration sets of q-ary symbols which get involved in the parity check equations. To do that, we start by selecting in each incoming message $\bar{U}_{pc}$ the n_q largest values (we will take n_q fixed in our simulation results for simplicity reasons) that we denote: $\bar{u}_{pc}^{(k_c)}$, k_c = 1,…,n_q. We use the following notation for the associated field element: $\alpha_c^{(k_c)}(x)$, so that we have:

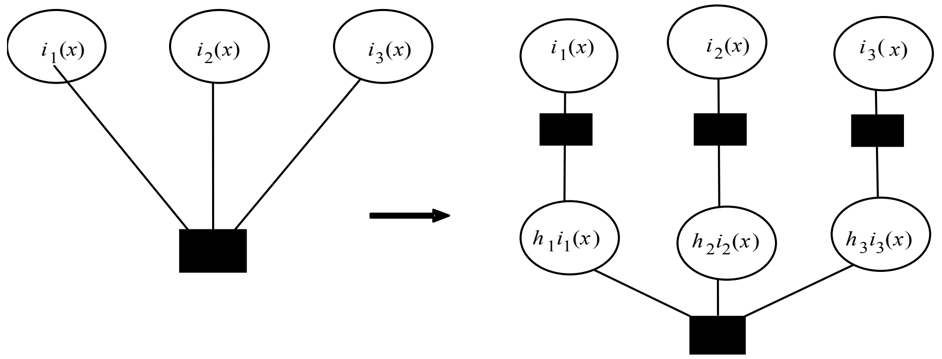

Figure 10. Transformation of the factor graph for the nonzero values in the *H*.

$$\bar{u}_{pc}^{(k_c)} = \log\left(\frac{\text{Prob}(h_c(x).i_c(x) = \alpha_c^{(k_c)}(x))}{\text{Prob}(h_c(x).i_c(x) = 0)}\right) \quad (31)$$
$$= \bar{U}_{pc}[\alpha_{c_1}^{(k_c)} \ldots \alpha_{c_m}^{(k_c)}]$$

With these largest values, we build the following set of configurations:

$$\text{Conf}(n_q) =$$
$$\left\{\boldsymbol{\alpha}_k = [\alpha_1^{(k_1)}(x), \ldots, \alpha_{d_c-1}^{(k_{d_c-1})}(x)]^T : \forall \boldsymbol{k} = [k_1, \ldots, k_{d_c-1}] \in \{1, \ldots, n_q\}^{d_c-1}\right\} \quad (32)$$

any vector of d_c-1 field elements in this set is called a configuration. The set Conf(n_q) corresponds to the set of configurations built from the n_q largest probabilities in each incoming message. Its cardinality is:

$$\text{Conf}(n_q) = n_q^{d_c-1}.$$

we need to assign a reliability to each configuration; we take as in [28]:

$$L(\boldsymbol{\alpha}_k) = \sum_{c=1 \ldots d_c-1} \bar{u}_c^{(k_c)}.$$

The initialization of the decoder is achieved with the channel log-likelihoods defined as: $L[i_1,\ldots,i_m]$.

The EMS proceeds in three steps as given below:

Sum-step**:** variable node update for a degree d_v node:

$$\bar{U}_{tp} = \bar{L} + \sum_{\text{v}=1, \text{v} \neq t}^{d_\text{v}} V_{p\text{v}} \quad t = 1, \ldots, d_\text{v} \quad (33)$$

$$\bar{U}_{tp}[i_1, \ldots, i_m] = \bar{L}[i_1, \ldots, i_m] +$$
or: $$\sum_{\text{v}=1, \text{v} \neq t}^{d_\text{v}} V_{p\text{v}}[i_1, \ldots, i_m] \quad (i_1, \ldots, i_m) \in \{0,1\}^m$$

Permutation-step**:** from variable to check nodes:

$$\bar{U}_{pc}[i_1, \ldots, i_m] =$$
$$\bar{U}_{\text{v}p}[j_1, \ldots, j_m] \quad (i_1, \ldots, i_m) \in \{0,1\}^m \quad (34)$$
with $i(x) = h(x).j(x)$

The permutation step from check to variable nodes is performed using: $P_{h(x)}^{-1}$.

Message update**:** for a degree d_c check node:

From the ***d*****$_c$-1** incoming messages $\bar{U}_{pc}$, build the sets:

$$S_{i_{d_c}(x)} = \text{Conf}_{i_{d_c}(x)}(q,1) \cup \text{Conf}_{i_{d_c}(x)}(n_p, n_c),$$

Then:

$$V_{d_c p}[i_{d_{c_1}}, \ldots, i_{d_{c_p}}] =$$
$$\max_{\boldsymbol{\alpha}_k \in S_{i_{d_c}(x)}} \{L(\boldsymbol{\alpha}_k)\} \quad (i_{d_{c_1}}, \ldots, i_{d_{c_p}}) \in \{0,1\}^m \quad (35)$$

With:

$$\text{Conf}_{i_{d_c}(x)}(n_p, n_c) =$$
$$\{\boldsymbol{\alpha}_k \in \text{Conf}(n_p, n_c) : h_{d_c}(x).i_{d_c}(x) + \sum_{c=1}^{d_c-1} \alpha_c^{(k_c)}(x) = 0\}$$

Post-processing:

$$\bar{V}_{cp}[i_1, \ldots, i_m] = \bar{V}_{cp}[i_1, \ldots, i_m] - \bar{V}_{cp}[0, \ldots, 0]$$
$$(i_1, \ldots, i_m) \in \{0,1\}^m, c = 1, \ldots, d_c \quad (36)$$

3.4. Simulation Results

We give here some simulation results to show the performance of the proposed scheme. Since the target comparison is the work of Kozic & al [27], we use their results as benchmark for our proposed system. In his work, Kozic plots the obtained BER results for n = 512 and 1024 as the size of the vector input bits together with Q = 2 or 3 and a random sparse matrix with weight ρ = 3 or 6 on each column. We take each time the best performances he obtained.

Using the same size of input blocks, we have to choose the desired spectral efficiency. Due to the heavy computational task, we only take here: q = 4 and q = 8; *i.e.*; we work with *GF*(4) and *GF*(8) with spectral efficiencies respectively equal to 2 and 3 bits/s/Hz. The obtained results are drawn on **Figure 11** for block size 512 and on **Figure 12** with block size 1024.

One notices that the performances of our *GF*(q) LDPC codes of size 512 are quite similar to those of Vucetic & al for size 1024 and, using block size of length 1024, our designs outperform clearly those of Vucetic & al. The SNR gain for block size 1024 is approximately equal to 0.5 dB for Q = 2 and for *GF*(8) and becomes 0.75 dB for Q = 2 and for *GF*(4). The improvement is slightly better in the case Q = 3 since we obtain gains of 1.0 dB for *GF*(8) and 1.5 dB for *GF*(4). This result is not really surprising since many authors have shown that LDPC codes over *GF*(q) exhibit better performances than their counterparts on *GF*(2). One can notice that the slopes *i.e.*; the diversity gain are the same each time in the waterfall region. A more detailed study should be done to determine the starting point SNR of the waterfall region with the EXIT-CHART curves.

4. Conclusions

In this paper we have proposed new insights of the work of Kozic *et al.* in the field of channel coding using chaos-based encoding process. First, using non-linear MOD-MAP mapping with matrices of small dimension, we are able, thanks to an original EM based guessing algorithm, to optimize the distance spectrum of the corresponding Chaos Coded Modulation (CCM) schemes and

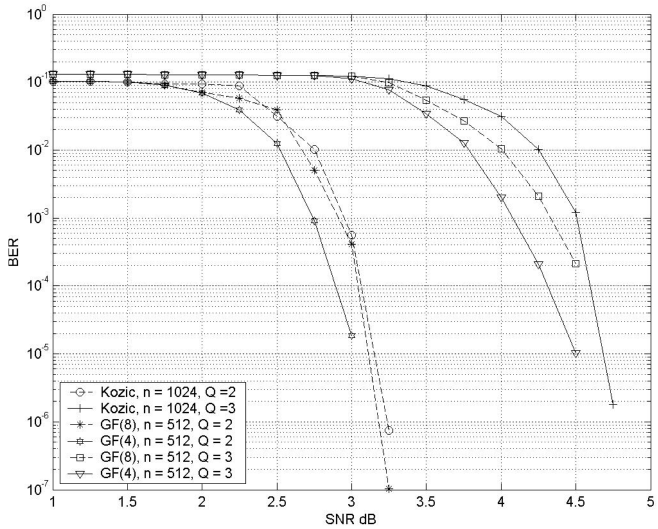

Figure 11. BER performances for chaos based LDPC codes over *GF*(*q*); block-length size equal to: 512.

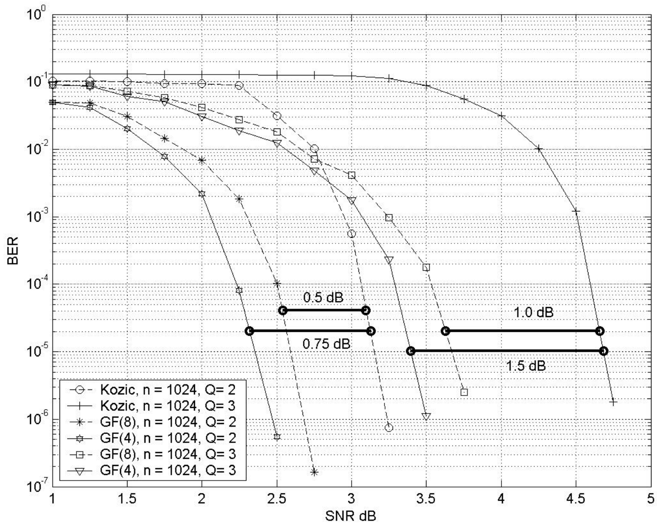

Figure 12. BER performances for chaos based LDPC codes over *GF*(*q*); block-length size equal to: 1024.

hence we can optimize the BER performances of such schemes. In the case where we use high dimensional sparse matrices for the MOD-MAP mapping, we can use a decoding process similar to those of LDPC codes. When we compare our obtained results with those of the former literature we noticed that, working on *GF*(*q*) enables to obtain significant gains of approximately 1. dB. This encouraging result entails the necessity to further optimize the design to reduce the hardware complexity. In fact, despite the use of the Extended Min-Sum algorithm, the obtained coding structure remains of prohibitive complexity.

5. References

[1] D. R. Frey, "Chaotic Digital Encoding: An Approach to Secure Communication," *IEEE Transactions on Circuits and Systems* II, Vol. 40, No. 10, 1993, pp. 660-666.

[2] A. Layec, "Développement de modèles de CAO pour la simulation système des systèmes de communication. Application aux communications chaotiques," Ph.D. Thesis, Xlim University of Limoges, 2006.

[3] F. J. Escribano, L. Lopez, M. A. F. Sanjuan, "Evaluation of Channel Coding and Decoding Algorithms Using Discrete Chaotic Maps," Chaos, American Institute of Physics, No. 16, 2006, pp. 1054-1500.

[4] I. P. Marino, L. Lopez, M. A. F. Sanjuan, "Channel Coding in Communications Using Chaos," *Physics Letters Elsevier Science*, Vol. 295, No. 4, 2002, pp. 185-191.

[5] E. Bollt, Y. C. Lai, C. Grebogi, "Channel Channel Capacity and Noise Resistance in Communicating with

Chaos," *Physical Review Letters American Physical Society*, Vol. 79, No. 19, pp. 3787-3790.

[6] T. Schimming and M. Hasler, "Coded Modulation Based On Controlled 1-D and 2-D Piecewise Linear Chaotic Maps," *IEEE International Symposium on Circuits and Systems* (*ISCAS*), pp. 762-765.

[7] B. Chen and G. W. Wornel, "Analog Error Correcting Codes Based on Chaotical Dynamical Systems," *IEEE Transactions on Communications*, Vol. 46, No. 7, pp. 881-890.

[8] T. Erseghe and N. Bramante, "Pseudo Chaotic Encoding Applied to Ultra-Wide-Band Impulse Radio," *IEEE Vehicular Technology Conference*, Vancouver, September 2002, pp. 1711-1715.

[9] J. Lee, C. Lee and D. B. Williams, "Secure Communications Using Chaos," *IEEE Globecom*'95, Singapore, November 1995, pp. 1183-1187.

[10] J. Lee, S. Choi and D. Hong, "Secure Communications Using a Chaos System in a Mobile Channel," *IEEE Globecom*'98, Sydney, November 1998. pp. 2520-2525.

[11] C. Lee and D. B. Williams, "A Noise Reduction Method for Chaotic Signals," 44*th IEEE Acoustic, Sensor and Signal Processing Conference* (*ICASSP*), Detroit, May 1995, pp. 1348-1351.

[12] A. M. Guidi, "Turbo and LDPC Coding for the AWGN and Space-Time Channel," Ph.D. Thesis, University of South Australia, South Australia, June 2006.

[13] S. A. Barbulescu, A. Guidi and S. S. Pietrobon, "Chaotic Turbo Codes," *IEEE Conference International Symposium on Information Theory* (*ISIT*), June 2000, p. 123.

[14] X. L. Zhou, J. B. Liu, W. T. Song and H. W. Luo, "Chaotic Turbo Codes in Secure Communications," *Conference EUROCON*'2001*, International Conference on Trends in Communications*, July 2001, pp. 199-201.

[15] A. Abel and W. Schwarz, "Chaos Communications-Principles, Schemes, and System Analysis," *Proceedings of the IEEE*, Vol. 90, No. 5, May 2002, pp. 691-710.

[16] W. M. Tam, F. C. M. Lau and C. K. Tse, "Digital Communications with Chaos," Elsevier, Oxford, 2007.

[17] S. Kozic, K. Oshima and T. Schimming, "How to Repair CSK Using Small Perturbation Control-Case Study and Performance Analysis," *Proceedings of ECCTD*, Krakow, August 2003.

[18] Y. S. Lau, Z. M. Hussain, "A New Approach in Chaos Shift Keying for Secure Communication," *Proceedings of the third International Conference on Information Technology and Applications* (*ICITA*'05), Sydney, pp. 630-633.

[19] H. Dedieu, M. P. Kennedy and M. Hasler, "Chaos Shift Keying: Modulation and Demodulation of a Chaotic Carrier Using Self-Synchronizing Chua's Circuits," *IEEE Transactions on Circuits and Systems* II: *Analog and Digital Signal Processing*, Vol. 40, No. 10, 1993, pp. 634-642.

[20] J. Schweitzer, "The Performance of Chaos Shift Keying: Synchronization Versus Symbolic Backtracking," *Proceedings of the IEEE International Symposium on Circuits and Systems* (*ISCAS*), Monterey, 1998, pp. 469-472.

[21] F. J. Escribano, L. Lopez and M. A. F. Sanjuan, "Parallel Concatenated Chaos Coded Modulations," *IEEE Conference* 15*th International Conference on Software*, *Telecommunications and Computer Networks*, Softcom, 2007.

[22] F. J. Escribano, L. Lopez and M. A. F. Sanjuan, "Iteratively Decoding Chaos Encoded Binary Signals," *Proceedings of the Eighth International Symposium on Signal Processing and its Applications*, 2005, Sydney, August 2005, pp. 275-278.

[23] S. Kozic, T. Schimming and M. Hasler, "Controlled One-and Multidimensional Modulations Using Chaotic Maps," *IEEE Transactions on Circuits and Systems* I: *Regular Papers*, Vol. 53, No. 9, September 2006, pp. 2048-2059.

[24] F. J. Escribano, L. Lopez and M. A. F. Sanjuan, "Chaos-Coded Modulations Over Rician and Rayleigh Flat Fading Channels", *IEEE Transactions on Circuits and Systems* II, Vol. 55, No. 6, June 2008, pp. 581-585**.**