Nonbinary Hybrid LDPC Codes

Lucile Sassatelli, David Declercq

Non-binary Hybrid LDPC Codes

Lucile Sassatelli and David Declercq
ETIS - CNRS UMR 8051 - ENSEA - University of Cergy-Pontoise, France

Abstract

In this paper, a new class of LDPC codes, named hybrid LDPC codes, is introduced. Hybrid LDPC codes are characterized by an irregular connectivity profile and heterogeneous orders of the symbols in the codeword. We show in particular that the class of hybrid LDPC codes can be asymptotically characterized and optimized using density evolution (DE) framework, and we also present a technique to maximize the minimum distance of the code. Numerical assessment of hybrid LDPC code performances is provided, by comparing them to protograph-based and multi-edge type LDPC codes. We show that hybrid LDPC codes allow to achieve an interesting trade-off between good error-floor performance and good waterfall region with non-binary coding techniques.

I. Introduction

During the 1990s, remarkable progress was made towards the Shannon limit, using codes that are defined in terms of sparse random graphs, and which are decoded by a simple probability-based message-passing algorithm. Two families of sparse-graph codes are excellent for error-correction: Low-Density Parity-Check (LDPC) codes, and Turbo Codes. The class of LDPC codes was first proposed in [1] in 1963, and rediscovered thirty years later [2], [3], [4], [5], after the invention of Turbo Codes [6]. LDPC codes are decoded through the iterative local message-passing algorithm based on the *Belief Propagation* (BP) principle [7]. These codes have been shown to exhibit excellent performance, under iterative BP decoding, over a wide range of communication channels, approaching channel capacity with moderate decoding complexity.

Asymptotically in the codeword length, LDPC codes exhibit a threshold phenomenon. Indeed, if the noise level is smaller than a certain decoding threshold (which depends on the bipartite graph properties) then it is possible to achieve an arbitrarily small bit error probability under iterative decoding, as the codeword length and the number of decoding iterations tend to infinity. On the contrary, for noise level larger than the threshold, the bit error probability is always larger than a positive constant, for any codeword length [4], [5]. There are two main tools for asymptotic analysis of LDPC codes, i.e. for

evaluating the decoding threshold associated to a given degree distribution: density evolution [4] and EXtrinsic Information Transfer (EXIT) charts [8]. One of the features that makes LDPC codes very attractive is the possibility to design, for several transmission channels, the degree distribution of the bipartite graph which provides a decoding threshold extremely close to the channel capacity [9].

While the asymptotic analysis and design of LDPC codes is mostly understood, the design of finite-length LDPC codes still remains an open question. Indeed, the local message-passing algorithm corresponds to the exact computation of *a posteriori* probabilities of variable values only if the graph is cycle-free, i.e., the BP decoder is exactly the Maximum-Likelihood (ML) decoder because it finds the global maximum of the ML criterion. In the finite length case, cycles appear in the graph [10]. In that case, the BP decoder does not compute anymore the *a posteriori* probabilities of variable values, thereby turning into suboptimal in the sense it does not correspond anymore to ML decoding. This leads to the loss of performance of BP decoding, compared to ML decoding, and particularly in the error-floor region. Moreover, finite length LDPC codes with a degree distribution associated to a decoding threshold close to capacity, though characterized by very good waterfall performance, usually exhibit bad error floor performance. This is due to a large fraction of degree-2 variable nodes leading to a poor minimum distance [11], [12].

The attempt to improve the trade-off between waterfall performance and error floor has recently inspired the study of more powerful, and somewhat more complex, coding schemes. This is the case of non-binary LDPC codes, Generalized LDPC (GLDPC) codes [13], Doubly-Generalized LDPC (D-GLDPC) codes [14] or Tail-biting LDPC (TLDPC) codes [15]. Non-binary LDPC codes have been introduced by Gallager in [1], and their finite-length assets have been underlined by Davey *et al.* in [16]. The main interest of non-binary LDPC codes actually lies in the decoder: good non-binary LDPC codes have much sparser factor graphs (or Tanner graphs) than binary LDPC codes [17], and the BP decoder is closer to optimal decoding since the small cycles can be avoided with a proper graph construction, as proposed in [18].

In order to improve the trade-off between waterfall performance and error floor, we introduce and study a new class of LDPC codes that we call *hybrid LDPC codes*. The class of hybrid LDPC codes is a generalization of existing classes of LDPC codes, both binary and non-binary. For hybrid LDPC codes, we allow the connectivity profile to be irregular and the orders of the symbols in the codeword to be heterogeneous. The rest of the paper is organized as follows. In Section II, notation is given. The structure and decoding of the class of hybrid LDPC codes are given in Section III. The asymptotic analysis is presented in Section IV, and the distribution optimizations in Section V. Section VI presents a finite-length optimization of hybrid LDPC codes, and Section VII some numerical results. The proofs

are gathered in the Appendix.

II. NOTATION

Vectors are denoted by boldface notations, e.g., $\mathbf{x}$. Random variables are denoted by upper-case letters, e.g. X and their instantiations in lower-case, e.g., x. There are two possible representations for the messages: plain-density probability vectors or Log-Density-Ratio (LDR) vectors. We denote the q elements of the finite group $G(q)$ (or the finite field $GF(q)$), of order q, by $(\alpha_0, \ldots, \alpha_{q-1})$, where $\alpha_0 = 0$. The probability that the random variable X takes the value x is denoted by $P(X = x)$. A q-dimensional probability vector is a vector $\mathbf{x} = (x_0, \ldots, x_{q-1})$ of real numbers such that $x_i = P(X = \alpha_i)$ for all i, and $\sum_{i=0}^{q-1} x_i = 1$.

Definition 1: Given a probability vector $\mathbf{x}$, the components of the corresponding LDR vector $\mathbf{w}$ are defined as

$$w_i = \log\left(\frac{x_0}{x_i}\right), \ i = 0, \ldots, q-1 \ .$$

The natural logarithm is used. We use the notation $\mathbf{w} = LDR(\mathbf{x})$. Note that for all $\mathbf{x}$, $w_0 = 0$. We define the LDR-vector representation of $\mathbf{x}$ as the $q-1$ dimensional vector $\mathbf{w} = (w_1, \ldots, w_{q-1})$. The observation of the channel under LDR form is a Logarithmic Likelihood Ratio (LLR). For convenience, in the derivation of the message properties and the corresponding proofs, the value $w_0 = 0$ is not defined as belonging to $\mathbf{w}$. Given an LDR-vector $\mathbf{w}$, the components of the corresponding probability vector $\mathbf{x}$ can be obtained by

$$x_i = \frac{e^{-w_i}}{1 + \sum_{k=1}^{q-1} e^{-w_k}}, \ i = 0, \ldots, q-1 \ . \tag{1}$$

We use the notation $\mathbf{x} = LDR^{-1}(\mathbf{w})$. A random probability-vector is defined to be a q-dimensional random variable $\mathbf{X} = (X_0, ..., X_{q-1})$. A random LDR-vector is a $(q-1)$-dimensional random variable $\mathbf{W} = (W_1, ..., W_{q-1})$. We give the definition of the $+g$ operation, as introduced in [19]. Given a probability vector $\mathbf{x}$ and an element $g \in G(q)$, $\mathbf{x}^{+g}$ is defined by

$$\mathbf{x}^{+g} = (x_g, x_{1+g}, \ldots, x_{(q-1)+g})$$

where addition is performed over $G(q)$.

$\mathbf{x}^\star$ is defined as the set

$$\mathbf{x}^\star = \{\mathbf{x}, \mathbf{x}^{+1}, \ldots, \mathbf{x}^{+(q-1)}\} \ .$$

Moreover, $n(\mathbf{x})$ is defined as the number of elements $g \in G(q)$ satisfying $\mathbf{x}^{+g} = \mathbf{x}$.

The LDR vectors corresponding to $\mathbf{x}$ and $\mathbf{x}^{+g}$ are denoted by $\mathbf{w}$ and $\mathbf{w}^{+g}$, respectively. Owing to

Definition 1 of the components of a LDR vector, the i^{th} component of $\mathbf{w}^{+g}$ is w_i^{+g} which is defined by

$$w_i^{+g} = w_{g+i} - w_g, \quad \forall i = 0 \dots q-1 \ . \tag{2}$$

For the sequel, we simplify the notation as follows: for any group $G(q)$, for all $a \in \{0, \dots, q-1\}$, the element α_a is now denoted by a.

III. The class of hybrid LDPC codes

A. General hybrid parity-check equations

Classically, non-binary LDPC codes are described thanks to the local constraints given by parity-check equations involving some of the codeword symbols c_i. If a code is linear over a finite field $GF(q)$, the parity-check equation corresponding to the i^{th} row of the parity-check matrix $\mathbf{H}$, is

$$\sum_j h_{ij} c_j = 0 \quad \text{in } GF(q) \ . \tag{3}$$

The field $GF(2^p)$ can be represented using the vector space $\left(\frac{\mathbb{Z}}{2\mathbb{Z}}\right)^p$ in a natural way. Multiplications in $GF(2^p)$ can be represented as matrix multiplications, after choosing a suitable representation. The set of matrices representing field elements then forms a field of invertible matrices. Thus, interpreting variables as elements of $\left(\frac{\mathbb{Z}}{2\mathbb{Z}}\right)^p$ and using matrix multiplication to form linear constraints can be used to model LDPC over $GF(2^p)$. In all this work, p does not need to be prime.

We aim at generalizing the definition of the parity-check equation by allowing more general operations than multiplications by $h_{ij} \in GF(q)$, and moreover, by considering parity-checks where codeword symbols can belong to different order finite sets: $c_j \in G(q_j)$. $G(q_j)$ is a finite set of order $q_j = 2^{p_j}$ with a group structure. Indeed, we will only consider groups of the type $G(q_j) = \left(\left(\frac{\mathbb{Z}}{2\mathbb{Z}}\right)^{p_j}, +\right)$ with $p_j = \log_2(q_j)$. Such a group corresponds to an ensemble of p_j-sized vectors whose elements lie in $\frac{\mathbb{Z}}{2\mathbb{Z}}$.

Let N be the codeword size. A hybrid LDPC code is defined on the group G which is the Cartesian product of the groups to which the codeword symbols belong:

$$G = G(q_1) \times \cdots \times G(q_N) \ .$$

Let q_j denote the group order of the j^{th} codeword symbol (either information or redundancy as the considered codes are systematic as described later in Section III-E). Such a group order is equivalently called the group order of the j^{th} variable node, or of the j^{th} column of $\mathbf{H}$. Let q_i denote the group order of the i^{th} redundancy codeword symbol. Such a group order is equivalently called the group order of the i^{th} check node, or of the i^{th} row of $\mathbf{H}$. The non-zero elements of the parity-check matrix are maps

which project a value in the column group (variable node group), onto a value in the row group (check node group, see Figure 1). This is achieved thanks to functions named h_{ij} such that

$$\begin{aligned} h_{ij} : G(q_j) &\rightarrow G(q_i) \\ c_j &\mapsto h_{ij}(c_j) \end{aligned}$$

Hence, a hybrid parity-check equation is given by

$$\sum_j h_{ij}(c_j) = 0 \quad \text{in } G(q_i) \ . \tag{4}$$

We notice that, in equation (3) as well as in equation (4), the additive group structure defines the local constraints of the code. Moreover, as mentioned in [5] and deeply studied in [20], the additive group structure has a Fourier transform, whose importance for the decoding is pointed out in Section III-F.

To sum up, the graph of a hybrid LDPC code is made of the following components. Variable nodes belong to different order groups, the messages going out of variable nodes are therefore of different sizes. On each edge, there is a general application from the group of the variable node to the group of the check node. The messages going into a given check node are therefore of the same size, and a hybrid check node is a usual non-binary parity check node.

Let us notice that this type of LDPC codes built on product groups has already been proposed in the literature [21][22], but no optimization of the code structure has been proposed and its application was restricted to the mapping of the codeword symbols to different modulation orders.

Since the mapping functions h_{ij} can be of any type, the class of hybrid LDPC codes is very general and includes classical non-binary and binary codes.

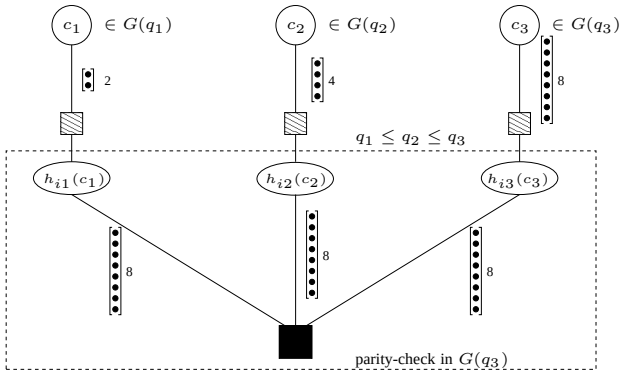

Fig. 1. Factor graph of parity-check of a hybrid LDPC code.

B. Different sub-classes of hybrid LDPC codes

Among the huge set of hybrid LDPC codes, we can distinguish as many classes as different types of non-zero elements of the parity-check matrix $\mathbf{H}$. As above mentioned, such a non-zero element is a map h_{ij}, which projects the q_j symbols of $G(q_j)$ onto a subset of q_j symbols of $G(q_i)$. Let us consider the case where these maps are linear, i.e., represented by a matrix, with dimensions $p_i \times p_j$. In that way, h_{ij} actually connects the binary map vector of a symbol in $G(q_j)$ to the binary map vector of a symbol in $G(q_i)$.

Remark: At this stage, it is quite straightforward to establish a connection between hybrid LDPC codes and Doubly-Generalized LDPC (D-GLDPC) codes, thoroughly studied in [14][23]. Indeed, the linear map h_{ij} can be seen as part of the generalized check node and generalized variable node. The code corresponding to the j^{th} generalized variable node v would have a number of information bits $K = p_j$ and length $N = \sum_i p_i$, where the sum is done over the groups of all the check nodes connected to v. The code of the i^{th} generalized check node c would have a number of redundancy bits $M = p_i$ and length $N = \sum_j p_j$, where the sum is done over the groups of all the variable nodes connected to c. However, it is important to note that, if the idea is the same, hybrid LDPC codes are not exactly D-GLDPC codes owing to the decoder. Indeed, with D-GLDPC codes, one considers that the generalized codes are at variable and check node sides, whereas with hybrid LDPC, we consider that the generalized codes for each node are split on the edges going into the node. As detailed in Section V on optimization, this difference allows us to affect different connection degrees on the nodes depending on their group order, i.e., depending on K for variable nodes and on M for check nodes. In other words, we will be able to optimize the length of the codes, given the dimension.

We distinguish different sub-classes of hybrid LDPC codes whose non-zero elements are linear maps:

(i) Maps which are not of rank p_j. This encompasses the case where the group order of a column is higher than the group order of the row. From a D-GLDPC perspective, this allows to have generalized variable nodes whose codes have $K > N$, that is to say the number of incoming bits is projected to a smaller one. This could be thought of as puncturing, and, as a consequence, we get back the result that the rate of the graph can be lower than the code rate. This case is out of the scope of this paper.

(ii) Maps which are of rank p_j. They are referred to as full-rank transforms, and correspond to matrices of size $p_i \times p_j$ with necessarily $p_j \leq p_i$. Such a map is depicted in Figure 2. We consider only these types of hybrid LDPC codes in this work.

C. Hybrid LDPC codes with linear maps

Let us consider all (i, j) such that there is an edge between the j^{th} variable node and the i^{th} check node. The corresponding column is in $G(q_j)$ and the corresponding row is in $G(q_i)$. In this work, we consider only hybrid LDPC codes whose non-zero elements are linear full-rank transforms of rank equal to $\log_2(q_j)$, thus with $q_j \leq q_i$. Linear maps always associate the null element of one group to the null element of the other. When looking at the factor graph of a hybrid LDPC code (see Figure 1), we

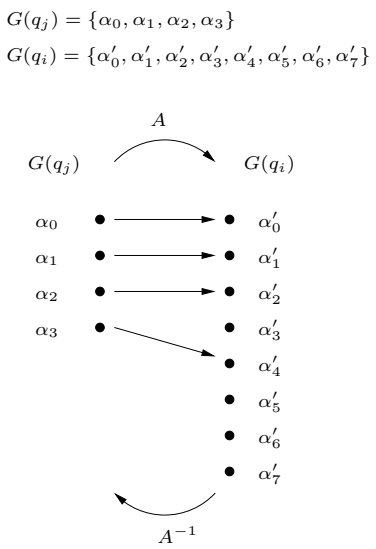

Fig. 2. Message transform through linear map.

note that an edge of the graph carries two kinds of probability-vector messages: messages of size q_j and messages of size q_i. Let A be an element of the set of linear maps from $G(q_j)$ to $G(q_i)$ which are full-rank. The transform of the probability vector is denoted *extension* from $G(q_j)$ to $G(q_i)$ when passing through A from variable node to check node, and the transform from $G(q_i)$ to $G(q_j)$ is denoted *truncation* from check node to variable node. Let Im(A) denote the image of A (A is injective since dim(Im(A))=rank(A)=p_j). The notations are the ones of Figure 2.

$$\begin{aligned} A: \qquad G(q_j) &\rightarrow G(q_i) \\ \alpha_k &\mapsto \alpha'_l = A(\alpha_k) \end{aligned}$$

Definition 2: The extension $\mathbf{y}$ of the probability vector $\mathbf{x}$ by A is denoted by $\mathbf{y} = \mathbf{x}^{\times A}$ and defined

by, for all $l = 0, \ldots, q_i - 1$,

$$y_l = \begin{cases} 0 & \text{if } \alpha'_l \notin \text{Im}(A); \\ x_k & \text{with } k \text{ such that } \alpha'_l = A(\alpha_k), \text{ if } \alpha'_l \in \text{Im}(A). \end{cases}$$

Although A is not bijective, we define A^{-1} by

$$\begin{aligned} A^{-1}: \qquad \text{Im}(A) &\rightarrow G(q_j) \\ \alpha'_l &\mapsto \alpha_k \text{ with } k \text{ such that } \alpha'_l = A(\alpha_k) \end{aligned}$$

Definition 3: The truncation $\mathbf{x}$ of the probability vector $\mathbf{y}$ by A^{-1} is denoted by $\mathbf{x} = \mathbf{y}^{\times A^{-1}}$ and defined by, for all $k = 0, \ldots, q_j - 1$,

$$x_k = y_l \text{ with } l \text{ such that } \alpha'_l = A(\alpha_k) \ .$$

It is worth noting that a vector resulting from truncation of a probability vector is not anymore a probability vector because a normalization would be needed to be so. When the decoding is performed using probability vectors instead of LDR vectors, we assume that only one normalization is performed at the end of the variable node update.

In the sequel, we use a shortcut by calling extension a linear map A, and by truncation A^{-1}. Indeed, extension or truncation are generated by a linear map A and do not apply to group elements, but to probability vectors. Additionally, we denote by $E_{j,i}$ the set of extensions from $G(q_j)$ to $G(q_i)$, and by $T_{i,j}$ the set of truncations from $G(q_i)$ to $G(q_j)$.

D. Parametrization of hybrid LDPC code ensembles

Classical LDPC codes are parametrized by two polynomials $(\lambda(x), \rho(x))$, whose each coefficient λ_i (resp. ρ_j) describes the fractions of edges connected to a variable node of degree i (resp. to a check node of degree j) [4]. Kasai et al. [24] introduced a detailed representation of LDPC codes, described by two-dimensional coefficients $\Pi(i, j)$, which are the fraction of edges connected to a variable node of degree i and also to a check node of degree j. Another detailed and more general representation of LDPC codes is the multi-edge type [25].

In our case, an edge of the Tanner graph has four parameters (i, q_k, j, q_l). We extend the notation adopted by Kasai et al. in [24], and we denote by $\Pi(i, j, k, l)$ the fraction of edges connected to a variable node of degree i in $G(q_k)$ and to a check node of degree j in $G(q_l)$.

Hence, $\Pi(i,j,k,l)$ is a joint probability which can be decomposed in several ways thanks to Bayes rule. For example, we have :

$$\Pi(i,j,k,l) \quad = \quad \Pi(i,j)\Pi(k,l|i,j)$$

where $\Pi(i,j)$ corresponds exactly to the definition adopted by Kasai, and $\Pi(k,l|i,j)$ describes the way the different group orders are allocated to degree i variable nodes and degree j check nodes.

An ensemble of hybrid LDPC codes is parametrized by Π and made of all the possible parity-check matrices whose parameters are those of the ensemble. The linear map of the parity-check matrices are chosen uniformly at random.

In the sequel, for more readable notations, we will write $\Pi(i,j,k)$ to denote the marginal distribution over l. The same with any other combinations of i,j,k,l, we will always use the same letters i,j,k,l to identify the parameters and the considered marginals.

Remark: Compared to D-GLDPC, the parametrization of hybrid LDPC codes allow to optimize the length of the generalized codes, both at variable and check nodes, given their dimensions K or M which are the group order characteristics. However, this representation is not as general as the one of multi-edge type LDPC codes [25] because it cannot distinguish a check node connected to only one degree-1 variable node, thereby preventing the use of degree-1 variable nodes in such described hybrid LDPC code ensembles.

We also define node wise fractions: $\tilde{\Pi}(i,k)$ and $\tilde{\Pi}(j,l)$ are the fractions of variable nodes of degree i in $G(q_k)$ and check nodes of degree j in $G(q_l)$, respectively. The connections between edgewise and nodewise fractions are the following:

$$\begin{aligned}
\tilde{\Pi}(i,k) &= \frac{\frac{\sum_{j,l}\Pi(i,j,k,l)}{i}}{\sum_{i,k}\frac{\sum_{j,l}\Pi(i,j,k,l)}{i}} \; ; \\
\tilde{\Pi}(j,l) &= \frac{\frac{\sum_{i,k}\Pi(i,j,k,l)}{j}}{\sum_{j,l}\frac{\sum_{i,k}\Pi(i,j,k,l)}{j}} \; .
\end{aligned} \tag{5}$$

The design code rate, i.e., the code rate when the parity-check matrix is full-rank, is expressed by:

$$R = 1 - \frac{\sum_l \left(\sum_j \frac{\sum_{i,k}\Pi(i,j,k,l)}{j}\right)\log_2(q_l)}{\sum_k \left(\sum_i \frac{\sum_{j,l}\Pi(i,j,k,l)}{i}\right)\log_2(q_k)} \; .$$

We define the *graph rate* as the rate of the binary LDPC code whose Tanner graph has parameters $\Pi(i,j)$. It is interesting to express the graph rate R_g in terms of Π, to compare it to the code rate of the hybrid

code:

$$R_g = 1 - \frac{\sum_j \frac{\sum_i \Pi(i,j)}{j}}{\sum_i \frac{\sum_j \Pi(i,j)}{i}} \ .$$

For the linear maps we consider, variable nodes are always in groups of order lower than or equal to the group orders of the check nodes to which they are connected. Hence the graph rate will be always higher than the code rate.

E. Encoding of hybrid LDPC codes

To encode hybrid LDPC codes whose non-zero elements are aforementioned full-rank linear maps, we consider upper-triangular parity-check matrices which are full-rank, i.e., without all-zero rows. The redundancy symbols are computed recursively, starting from the redundancy symbol depending only on information symbols. The images by the linear maps of the symbols involved in the parity-check equation but the redundancy symbol being computed, are summed up. The summation is performed in the group of the redundancy symbol, i.e., the group of the corresponding row. The redundancy symbol is set to the inverse of this sum by the linear map connected to it. This linear map is bijective from $G(q_l)$ to $G(q_l)$, if $G(q_l)$ is the group the redundancy symbol belongs to. Hence, information symbols satisfy that any assignment of values to them is valid, and the redundancy symbols are computed from them.

F. Decoding algorithm for hybrid LDPC codes

To describe the BP decoding, let $\mathbf{l}_{cv}^{(t)}$ denote the message going into variable node v from check c at the t^{th} iteration, and $\mathbf{r}_{vc}^{(t)}$ the probability-vector message going out of variable node v to check node c at the t^{th} iteration. The connection degrees of v and c are denoted by d_v and d_c, respectively. Let A_{vc} denote the linear map on the edge connecting variable node v to check node c. The a^{th} component of $\mathbf{l}_{cv}^{(t)}$ is denoted by $l_{cv}^{(t)}(a)$. The same holds for $r_{vc}^{(t)}(a)$. Let $\mathbf{x}$ be the sent codeword and N the number of codeword symbols. We recall that we simplify the notation as follows: for any group $G(q)$, for all $a \in \{0, \ldots, q-1\}$, the element α_a is now denoted by a. Also, since A is a linear map, the matrix of the map is also denoted by A. Hence, for all linear maps A from $G(q_1)$ to $G(q_2)$, $A(\alpha_i) = \alpha_j$ with $\alpha_i \in G(q_1)$ and $\alpha_j \in G(q_2)$, is translated into $Ai = j$ with $i \in \{0, \ldots, q_1 - 1\}$ and $j \in \{0, \ldots, q_2 - 1\}$.

- Initialization: Let $x_i \in G(q_i)$ be the i^{th} sent symbol and y_i be the corresponding channel output, for $i = 0 \ldots N-1$. For each check node c connected to the v^{th} variable node v, and for any $a \in \{0, \ldots, q_k - 1\}$:

$$r_{vc}^{(0)}(a) = r_v^{(0)}(a) = P(Y_v = y_v | X_v = a) \ ;$$

$$l_{vc}^{(0)}(a) = 1 \ .$$

- Check node update: Consider a check node c and a variable node v. Let $\{v_1, \ldots, v_{d_c-1}\}$ be the set of all variable nodes connected to c, except v. Let G be the Cartesian product group of the groups of the variable nodes in $\{v_1, \ldots, v_{d_c-1}\}$. For all $a \in G(q_v)$

$$l_{cv}^{(t)}(a) = \sum_{\substack{(b_1,\ldots,b_{d_c-1}) \in G: \\ \bigoplus_{i=1}^{d_c-1} A_{v_i c} b_i = A_{vc} a}} \prod_{n=1}^{d_c-1} r_{v_n c}^{(t)}(b_n) \tag{6}$$

  where the $\bigoplus$ operator highlights that the addition is performed over $G(q_c)$, the group of the row corresponding to c, as defined in Section III-C.

- Variable node update: Consider a check node c and a variable node v. Let $\{c_1, \ldots, c_{d_v-1}\}$ be the set of all check nodes connected to v, except c. For all $a \in G(q_v)$

$$r_{vc}^{(t+1)}(a) = \mu_{vc} r_v^{(0)}(a) \prod_{n=1}^{d_v-1} l_{c_n v}^{(t)}(a) \tag{7}$$

  where μ_{vc} is a normalization factor such that $\sum_{a=0}^{q_v-1} r_{vc}^{(t+1)}(a) = 1$.

- Stopping criterion: Consider a variable node v. Let $\{c_1, \ldots, c_{d_v}\}$ be the set of all check nodes connected to v. Equation (8) corresponds to the decision rule on symbols values, at iteration t:

$$\hat{x}_v^{(t)} = \arg\max_a r_v^{(0)}(a) \prod_{n=1}^{d_v} l_{c_n v}^{(t)}(a) \ . \tag{8}$$

  Variable and check node updates are performed iteratively until the decoder has converged to a codeword, or until the maximum number of iterations is reached.

It is possible to have an efficient Belief propagation decoder for hybrid LDPC codes. As mentioned in [5][20], the additive group structure has a Fourier transform, so that efficient computation of the convolution can be done in the Fourier domain. One decoding iteration of BP algorithm for hybrid LDPC codes, in the probability domain with a flooding schedule, is composed of:

- Step 1 **Variable node update** in $G(q_j)$: pointwise product of incoming messages followed by a normalization
- Step 2 **Message extension** $G(q_j) \rightarrow G(q_i)$ (see Definition 2)
- Step 3 **Parity-Check update** in $G(q_i)$ in the Fourier domain
  - FFT of size q_i
  - Pointwise product of FFT vectors
  - IFFT of size q_i

- Step 4 **Message truncation** from $G(q_i) \rightarrow G(q_j)$ (see Definition 3)

We do not perform a detailed complexity analysis, but we provide the following discussion. Let us consider equations (7) and (6). In terms of number of operations per iteration, the complexity of hybrid LDPC decoding is upper-bounded by the complexity of decoding a non-binary LDPC code in the highest order field. Although the decoding complexity of hybrid LDPC codes is clearly higher than that of binary LDPC codes, it is worth noting that hybrid LDPC codes are compliant with reduced complexity non-binary decoders which have been presented recently in the literature [26], [27]. In particular, [26] introduces simplified decoding of $GF(q)$ LDPC codes and shows that they can compete with binary LDPC codes even in terms of decoding complexity.

IV. Asymptotic analysis of hybrid LDPC code ensembles

This section describes the density evolution analysis for hybrid LDPC codes. Density evolution is a method for analyzing iterative decoding of code ensembles. We first prove that, on a discrete memoryless symmetric-output channel, the analysis can be led assuming that the all-zero codeword is transmitted, because the error probability of the hybrid LDPC decoding is independent of the transmitted codeword.

We express the density evolution for hybrid LDPC codes, and mention the existence of fixed points, which can be used to determine whether or not the decoding of a given hybrid LDPC code ensemble is successful for a given SNR, in the infinite codeword length case. Thus, convergence thresholds of hybrid LDPC codes are similarly defined as for binary LDPC codes [4]. However, as for $GF(q)$ LDPC codes, the implementation of density evolution of hybrid LDPC codes is too computationally intensive, and an approximation is needed.

Thus, we derive a stability condition, as well as the EXIT functions of hybrid LDPC decoder under Gaussian approximation, with the goal of finding good parameters for having good convergence threshold.

A. Channel symmetry

Only memoryless symmetric channels are considered in this work. Extension to arbitrary memoryless channels can be done by a coset approach, as detailed in [19] for $GF(q)$ LDPC codes. In this section, we introduce classical results leading to asymptotic analysis, but we prove them in the specific case of hybrid LDPC codes and of the definition of channel symmetry we consider.

Definition 4: [28] A channel is symmetric when the density of the observation in probability form fulfills:

$$P(\mathbf{Y} = \mathbf{y}|x = i) = P(\mathbf{Y} = \mathbf{y}^{+i}|x = 0)$$

Lemma 1: Let $P_e^{(t)}(\mathbf{x})$ denote the conditional error probability after the t^{th} BP decoding iteration of a hybrid LDPC code, assuming that codeword $\mathbf{x}$ was sent. If the channel is symmetric, then $P_e^{(t)}(\mathbf{x})$ is independent of $\mathbf{x}$.

The proof of this lemma is provided in Appendix. This property allows to assume that the all-zero codeword has been transmitted, for the remaining of the asymptotic analysis of hybrid LDPC code ensemble performance.

B. Message symmetry

The channel symmetry can entail a certain property of messages spreading over the graph during the decoding iterations. This property is the symmetry of the messages. The definitions of symmetric probability vectors and LDR vectors are given hereafter.

Definition 5: [19] A random probability-vector $\mathbf{Y}$ is symmetric if for any probability vector $\mathbf{y}$, the following expression holds:

$$P(\mathbf{Y} = \mathbf{y} | \mathbf{Y} \in \mathbf{y}^{\star}) = y_0 \cdot n(\mathbf{y}) \tag{9}$$

where $\mathbf{y}^{\star}$ and $n(\mathbf{y})$ are as defined in Section II.

Definition 6: [19] Let $\mathbf{W}$ be a random LDR-vector. The random variable $\mathbf{Y} = LDR^{-1}(\mathbf{W})$ is symmetric when $\mathbf{W}$ satisfies

$$P(\mathbf{W} = \mathbf{w}) = e^{w_i} P(\mathbf{W} = \mathbf{w}^{+i}) \tag{10}$$

for all LDR vectors $\mathbf{w}$.

The proof of the equivalence between these two definitions is provided in [19].

Let us connect the channel symmetry property to the message symmetry property.

Lemma 2: Let $\mathbf{Y}$ be the observation in probability form, and let $\mathbf{W} = LDR(\mathbf{Y})$. If the channel is symmetric, then, under the all-zero codeword assumption, the density P_0 of $\mathbf{W}$ is symmetric:

$$P(\mathbf{W} = \mathbf{w} | x = 0) = P_0(\mathbf{w}) = e^{w_i} P_0(\mathbf{w}^{+i})$$

The proof of this lemma is provided in Appendix.

Lemma 3: If the bipartite graph of a hybrid LDPC code is cycle-free, then, under the all-zero codeword assumption, all the messages on the graph at any iteration of BP decoding, are symmetric.

Proof of Lemma 3 is given in Appendix.

C. Density evolution

Analogously to the binary or non-binary cases, density evolution for hybrid LDPC codes tracks the distributions of messages produced by the BP algorithm, averaged over all possible neighborhood graphs on which they are based. The random space is comprised of random channel transitions, the random selection of the code from a hybrid LDPC ensemble parametrized by Π, and the random selection of an edge from the graph. The random space does not include the transmitted codeword, which is assumed to be set to the all-zero codeword (following Lemma 1). We denote by $\mathbf{R}^{(k)^{(0)}}$ the initial message across an edge connected to a variable in $G(q_k)$, by $\mathbf{R}^{(i,k)^{(t)}}$ the message going out of a variable node of degree i in $G(q_k)$ at iteration t. The message going out of a check node of degree j in $G(q_l)$ at iteration t is denoted by $\mathbf{L}^{(j,l)^{(t)}}$. We denote by $\mathbf{x}_l$ and $\mathbf{x}_k$ any two probability vectors of size q_l and q_k, respectively.

Let us denote by $\mathcal{P}_q$ the set of all probability vectors of size q. Let $\mathbf{r}_{q_k}(\mathbf{r}^{(0)}, \mathbf{l}^{(1)}, ..., \mathbf{l}^{(i-1)})$ denote the message map of a variable node of degree i in $G(q_k)$, as defined in equation (7): the input arguments are i probability vectors of size q_k. Let $\mathbf{l}_{q_l}(\mathbf{r}^{(1)}, ..., \mathbf{r}^{(j-1)})$ denote the message map of a check node of degree j in $G(q_l)$: the input arguments are $j-1$ probability vectors of size q_l.

$$P(\mathbf{L}^{(j,l)^{(t)}} = \mathbf{x}_l) = \sum_{\substack{\mathbf{r}^{(1)},...,\mathbf{r}^{(j-1)} \in \mathcal{P}_{q_l}: \\ \mathbf{l}_{q_l}(\mathbf{r}^{(1)},...,\mathbf{r}^{(j-1)}) = \mathbf{x}_l}} \prod_{n=1}^{j-1} \sum_{i,k} \Pi(i,k|j,l) \sum_{\substack{A \in E_{k,l}: \\ (\mathbf{r}^{(n)^{\times A^{-1}}})^{\times A} = \mathbf{r}^{(n)}}} P(A) P(\mathbf{R}^{(i,k)^{(t)}} = \mathbf{r}^{(n)^{\times A^{-1}}}) \; ; \tag{11}$$

$$P(\mathbf{R}^{(i,k)^{(t)}} = \mathbf{x}_k) = \sum_{\substack{\mathbf{r}^{(0)},\mathbf{l}^{(1)},...,\mathbf{l}^{(i-1)} \in \mathcal{P}_{q_k}: \\ \mathbf{r}_{q_k}(\mathbf{r}^{(0)},\mathbf{l}^{(1)},...,\mathbf{l}^{(i-1)}) = \mathbf{x}_k}} P(\mathbf{R}^{(k)^{(0)}} = \mathbf{r}^{(0)}) \prod_{n=1}^{i-1} \sum_{j,l} \Pi(j,l|i,k) \sum_{A \in E_{k,l}} P(A) \sum_{\substack{\mathbf{r} \in P_{q_l}: \\ \mathbf{r}^{\times A^{-1}} = \mathbf{l}^{(n)}}} P(\mathbf{L}^{(j,l)^{(t)}} = \mathbf{r}) \; . \tag{12}$$

Richardson and Urbanke [5] proved a *concentration theorem* that states that, as the block length N tends to infinity, the bit error rate at iteration t, of any graph of a given code ensemble, converges to the probability of error on a cycle-free graph in the same ensemble. The convergence is in probability, exponentially in N. As explained in [19] for classical non-binary LDPC codes, this theorem carries over hybrid LDPC density-evolution unchanged by replacing bit- with symbol- error rate.

Moreover, one can prove that the error-probability is a non-increasing function of the decoding iterations, in a similar way to the proof of Theorem 7 in [4]. This non-increasing property ensures that the sequence corresponding to density evolution, by iterating between equations (11) and (12), converges to a fixed point. Implementing the density evolution allows to check whether or not this fixed point corresponds to the zero error probability, which means that the decoding in the infinite codeword length

case has been successful. Furthermore, Richardson and Urbanke proved in [5] the monotonicity of error probability in terms of the channel parameter for physically degraded channels. Thus hybrid LDPC codes, like binary or non-binary LDPC codes, exhibit a threshold phenomenon.

Like for $GF(q)$ LDPC codes, implementing the density evolution for hybrid LDPC codes is too computationally intensive. Thus, in the sequel, we present a useful property of hybrid LDPC code ensembles, which allows to derive both a stability condition and an EXIT chart analysis for the purpose of approximating the exact density evolution for hybrid LDPC code ensembles.

D. Invariance induced by linear maps (LM-invariance)

Bennatan and Burshtein in [19] used permutation-invariance to derive a stability condition for non-binary LDPC codes, and to approximate the densities of graph messages using a one-dimensional parameter. The difference between non-binary and hybrid LDPC codes lies in the non-zero elements of the parity-check matrix. Indeed, the non-zero elements do not correspond anymore to cyclic permutations, but to extensions or truncations (see Definitions 2 and 3). Our goal in this section is to prove that linear-map invariance (shortened by LM-invariance) of messages is induced by choosing uniformly the linear maps as non-zero elements.

Until the end of the current section, we work with probability domain random vectors, but all the definitions and proofs also apply to LDR random vectors. Let us recall that $E_{j,i}$ is the set of extensions from $G(q_j)$ to $G(q_i)$, and $T_{i,j}$ is the set of truncations from $G(q_i)$ to $G(q_j)$.

Definition 7: A random vector $\mathbf{Y}$ of size q_l is said to be LM-invariant when for all $k < l$ and $(A^{-1}, B^{-1}) \in T_{l,k} \times T_{l,k}$, the random vectors $\mathbf{Y}^{\times A^{-1}}$ and $\mathbf{Y}^{\times B^{-1}}$ are identically distributed, i.e., when $P(\mathbf{Y}^{\times A^{-1}} = \mathbf{y}) = P(\mathbf{Y}^{\times B^{-1}} = \mathbf{y})$ for all $\mathbf{y} \in \mathbb{R}^k$.

Lemma 4: If a random vector $\mathbf{Y}$ of size q_l is LM-invariant, then all its components are identically distributed.

Proof of Lemma 4 is given in Appendix.

Definition 8: Let $\mathbf{X}$ be a random vector of size q_k, we define the random-extension of size q_l of $\mathbf{X}$, denoted $\tilde{\mathbf{X}}$, as the random vector $\mathbf{X}^{\times A}$, where A is uniformly chosen in $E_{k,l}$ and independently of $\mathbf{X}$.

Lemma 5: Consider a random vector $\mathbf{Y}$ of size q_l. If there exist q_k and a random vector $\mathbf{X}$ of size q_k such that $\mathbf{Y} = \tilde{\mathbf{X}}$, then $\mathbf{Y}$ is LM-invariant.

Proof of Lemma 5 is given in Appendix.

Thanks to Lemma 7, the messages on the graph of a hybrid LDPC code, in the code ensemble with uniformly chosen extensions, are LM-invariant, except the messages going out of variable nodes.

E. The Stability condition for hybrid LDPC codes

The stability condition, introduced in [4], is a necessary and sufficient condition for the error probability to converge to zero, provided it has already dropped below some value. This condition must be satisfied by the Signal to Noise Ratio (SNR) corresponding to the threshold of the code ensemble. Therefore, ensuring this condition, when implementing an approximation of the exact density evolution, helps to have a more accurate approximation of the exact threshold.

In this paragraph, we generalize the stability condition to hybrid LDPC codes. Let $p(y|x)$ be the transition probabilities of the memoryless output symmetric channel and $c^{(k)}$ be defined by

$$c^{(k)} = \frac{1}{q_k - 1} \sum_{i=1}^{q_k - 1} \int \sqrt{p(y|i)p(y|0)} dy \ .$$

Let $\mathbf{x}$ be a positive real-valued vector of size the number of different group orders. Let us define the g function by:

$$g(k, c^{(k)}, \Pi, \mathbf{x}) = c^{(k)} \Pi(i = 2|k) \sum_{j,l} \Pi(j, l|i, k)(j - 1) \sum_{k'} \Pi(k'|j, l) \frac{q_{k'} - 1}{q_l - 1} x_{k'} \ .$$

For more readable notations, we also define the vector output function $\mathbf{G}(\mathbf{x})$ by:

$$\mathbf{G}(\mathbf{x}) = \{g(k, c^{(k)}, \Pi, \mathbf{x})\}_k$$

which means that the p^{th} component of $\mathbf{G}(\mathbf{x})$ is $G_p(\mathbf{x}) = g(p, c^{(p)}, \Pi, \mathbf{x})$. Let $P_e^{(k)^t} = P_e(\mathbf{R}_t^{(k)})$ be the probability that the message $\mathbf{R}_t^{(k)}$ be erroneous, i.e., corresponds to an incorrect decision. The average probability that any rightbound message be erroneous is $P_e^t = \sum_k \Pi(k) P_e^{(k)^t}$. Let us denote the convolution by $\otimes$. Then $\mathbf{x}^{\otimes n}$ corresponds to the convolution of vector $\mathbf{x}$ by itself n times.

Theorem 1: Consider a given hybrid LDPC code ensemble parametrized by $\Pi(i, j, k, l)$. If there exists a vector $\mathbf{x}$ with all positive components, such that, for all k,
$\lim_{n \to \infty} g(k, c^{(k)}, \Pi, \mathbf{G}^{\otimes n}(\mathbf{x})) = 0$, then there exist t_0 and ϵ such that, if $P_e^{t_0} < \epsilon$, then P_e^t converges to zero as t tends to infinity.

Proof of Theorem 1 is given in Appendix. This condition is sufficient for stability.

Let us note that, for a non-binary $GF(q)$ LDPC codes, the stability condition for hybrid LDPC codes reduces to the stability condition for $GF(q)$ LDPC codes, given by [19]. Indeed,

$$\lim_{n \to \infty} g(k, c^{(k)}, \Pi, \mathbf{G}^{\otimes n}(\mathbf{x})) = 0$$

is equivalent in this case to

$$\rho'(1)\lambda'(0) \frac{1}{q_k - 1} \sum_{i=1}^{q_k - 1} \int \sqrt{p(y|i)p(y|0)} dy < 1 \ .$$

When the transmission channel is BIAWGN, we have

$$\int \sqrt{p(y|i)p(y|0)}dy = exp(-\frac{1}{2\sigma^2}n_i) \ .$$

Let Δ be defined by

$$\Delta = \frac{1}{q_k - 1}\sum_{i=1}^{q_k-1} exp(-\frac{1}{2\sigma^2}n_i)$$

with n_i, the number of ones in the binary map of $\alpha_i \in G(q)$. Under this form, we can prove that Δ tends to zero as q goes to infinity on BIAWGN channel. This means that any fixed point of density evolution is stable as q tends to infinity for non-binary LDPC codes. This shows, in particular, that non-binary cycle-codes, i.e., with constant symbol degree $d_v = 2$, are stable at any SNR provided that q is large enough.

F. EXIT charts for hybrid LDPC codes

The purpose is to approximate the decoding threshold of a hybrid LDPC code ensemble with parameters Π, in such a way that it can be used in an optimization procedure, where the threshold will be used as the cost function. To do so, the message densities are projected on one-scalar parameter. The considered channel is the Binary Input Additive White Gaussian Noise (BIAWGN) channel with BPSK modulation.

With binary LDPC codes, Chung et al. [29] observed that the variable-to-check messages are well approximated by Gaussian random variables, in particular when the variable node degree is high enough. The approximation is much less accurate for messages going out of check nodes. Furthermore, the symmetry of the messages in binary LDPC decoding implies that the mean m and variance σ^2 of the random variable are related by $\sigma^2 = 2m$. Thus, a symmetric Gaussian random variable may be described by a single parameter. This property was also observed by ten Brink et al. [8] and is essential to their development of EXIT charts for Turbo Codes. In [14], the authors analysed D-GLDPC on the BEC, which allowed to track only one parameter, the extrinsic information, instead of complete message densities. In the context of non-binary LDPC codes, Li et al. [28] obtained a description of $q-1$-dimensional Gaussian distributed messages by $q-1$ parameters. Bennatan et al. in [19] used symmetry and permutation-invariance to reduce the number of parameters from $q-1$ to one. This enabled the generalization of EXIT charts to $GF(q)$ LDPC codes.

First, let us discuss the accuracy of the Gaussian approximation of the channel output in symbolwise LLR form for hybrid LDPC code ensembles. The channel outputs are noisy observations of bits, from which we obtain bitwise LLR, all identically distributed as $\mathcal{N}(\frac{2}{\sigma^2}, \frac{4}{\sigma^2})$ [29]. Let $\mathbf{s}$ be the vector gathering

the LLRs $b_1, \ldots, b_{p_k}$ of bits of which a symbol in $G(q_k)$ is made: $\mathbf{s} = (b_1, \ldots, b_{p_k})^T$. Each component of an input LLR random vector $\mathbf{l}$ of size $(q_k - 1)$ is then a linear combination of these bitwise LLRs:

$$\mathbf{l} = B_{q_k} \cdot \mathbf{s} \tag{13}$$

where B_{q_k} is the matrix of size $q_k \times \log_2(q_k)$ in which the i^{th} row is the binary map of the i^{th} element of $G(q_k)$. The distribution of initial messages is hence a mixture of one-dimensional Gaussian curves, but are not Gaussian distributed vectors. Indeed, it is easy to see that the covariance matrix of vector $\mathbf{l}$ is not invertible.

Secondly, let us introduce a slight extension of Theorem 6 in [19].

Theorem 2: Let $\mathbf{W}$ be an LDR random vector, Gaussian distributed with mean $\mathbf{m}$ and covariance matrix $\mathbf{\Sigma}$. Assume that the probability density function $f(\mathbf{w}$ of $\mathbf{W}$ exists and that $\mathbf{\Sigma}$ is nonsingular. Then $\mathbf{W}$ is both symmetric and LM-invariant if and only if there exits $\sigma > 0$ such that:

$$\mathbf{m} = \begin{bmatrix} \sigma^2/2 \\ \sigma^2/2 \\ \vdots \\ \sigma^2/2 \end{bmatrix} \quad , \quad \mathbf{\Sigma} = \begin{bmatrix} \sigma^2 & & & \sigma^2/2 \\ & \sigma^2 & & \\ & & \cdots & \\ \sigma^2/2 & & & \sigma^2 \end{bmatrix}$$

The proof of Theorem 2 is the same as the proof of Theorem 6 in [19], because the permutation-invariance property [19] is used only through the fact that the components of a vector satisfying this property are identically distributed. This fact is ensured by a LM-invariant vector thanks to Lemma 4.

Thirdly, Lemma 4 ensures that, if a vector is LM-invariant, then its components are identically distributed. Hence, if we assume that a message is Gaussian distributed, symmetric and LM-invariant, its density depends on only one-scalar parameter. Let us now discuss the relevance of approximating the message densities of a hybrid LDPC code ensemble by Gaussian random vectors. Let $\mathbf{r}^{(t)}(\mathbf{x})$ be the density of a LDR message going out of a variable node in $G(q_k)$ after being extended by an extension chosen uniformly at random in $E_{k,k}$. Any component of such vector has density $r^{(t)}(x)$. Messages going out of variable nodes are extended when passing through the linear extension function nodes. As described in Section III-C, the extension turns, e.g., a q_1-sized probability vector into a q_2-sized vector, with $q_2 \geq q_1$. This means that $q_2 - q_1$ of the resulting extended LDR message components are infinite, because these components of the corresponding probability vector are zero. Hence, the density of each component, of an extended message, is a mixture including a Dirac Δ_∞. Since this LDR vector is the random extension of the variable node output message, it is LM-invariant. From Lemma 4, each component is identically distributed.

Property 1: The probability density function of any component of an LDR message after extension at iteration t, is expressed as

$$d^{(t)}(x) = \beta r^{(t)}(x) + (1-\beta)\Delta_\infty$$

where the weight β is independent of t.

Proof: At any decoding iteration, $r^{(t)}(x)$ cannot have a Δ_∞ component because there exists no set of linear maps connected to the neighboring check nodes of v, such that there exists forbidden elements in $G(q_k)$ to which the symbol value associated to v cannot be equal. This is due to the fact that each check node (or the associated redundancy symbol) is in a group of order higher or equal to the group orders of its neighboring variable nodes. Hence, β is independent of the decoding iterations (it depends only on the groups of the codeword symbols).

□

It is therefore easy to show that any normalized moment, of order greater than 1, of the vector density (expectation of the product of a different number of its components) is equal to the same moment of the vector density $\mathbf{r}^{(t)}(\mathbf{x})$. Thus, if we assume that the vector density $\mathbf{r}^{(t)}(\mathbf{x})$, i.e., at variable node output, is dependent on only one scalar parameter, so is the whole density of the extended vector message. In other words, the density of vector message of a hybrid LDPC code cannot be approximated by a Gaussian density, owing to the Δ_∞ component in the density, but is dependent on only one parameter if we assume that the density $\mathbf{r}^{(t)}(\mathbf{x})$ is Gaussian. The same property holds for messages before truncation, if we assume that messages going into variable nodes are Gaussian distributed. Since the messages going into variable nodes are symmetric and LM-invariant, their sum done during the variable node update, is symmetric and LM-invariant by Lemma 18 in [19] and Lemma 9 (see Appendix). Hence, the one-scalar parameter approximation for hybrid LDPC codes is not less accurate than for $GF(q)$ LDPC codes [19].

The parameter, defining the message densities, we choose to track is the mutual information between a message and the codeword sent.

Definition 9: [19] The mutual information between a symmetric LDR-vector message $\mathbf{W}$ of size $q-1$ and the codeword sent, under the all-zero codeword assumption, is defined by:

$$I(C;\mathbf{W}) = 1 - \mathbb{E}\log_q\left(1 + \sum_{i=1}^{q-1} e^{-W_i} | C=0\right)$$

The equivalent definition for the probability vector $\mathbf{X} = LDR^{-1}(W)$ of size q is

$$I(C;\mathbf{X}) = 1 - \mathbb{E}\log_q\left(\frac{\sum_{i=0}^{q-1} X_i}{X_0} | C=0\right) . \tag{14}$$

In the following, the shortcut "mutual information of a LDR vector" is used instead of "mutual information between a LDR vector and the codeword sent".

Since the connection between mutual information and the expectation of a symmetric Gaussian distributed variable is easily obtained by interpolating simulation points, we consider expectations of Gaussian distributed vectors with same mutual information as the message vectors. That is we consider a projection of the message densities on Gaussian densities, based on Property 1 which ensures that densities of messages going out of or into check nodes are dependent on the same parameters as densities of messages going into or out of variable nodes. There are two models of messages handled by the hybrid decoder, and hence we define two functions to express the mutual information:

- Messages going out of variable nodes are not LM-invariant, and their mutual information is expressed thanks to a function called $J_v(\sigma^2, \mathbf{m}, q_k)$ in terms of the BIAWGN channel variance σ^2, a mean vector $\mathbf{m}$ and q_k, the group order of the variable node. The mean $\mathbf{m}$ is the mean of a Gaussian distributed vector.
- For a hybrid LDPC code ensemble with uniformly chosen linear maps, messages going into and out of check nodes are LM-invariant. If $G(q_l)$ denotes the group of the check node, the mutual information of messages is expressed by a function $J_c(m, q_l)$. m is the mean of a Gaussian random variable (any component of a Gaussian distributed vector with same mutual information as the graph message).

Let us now detail the evolution of mutual information of messages through BP decoding.

- The mutual information of a variable node output is expressed thanks to the $J_v(\cdot,\cdot,\cdot)$ function applied to sum of means, since variable node update is the summation of LDRs. Here, x_{in} is the mutual information of truncation operator output, and $\mathbf{1}_{q_k}$ is the all-one vector of size q_k. The mutual information x_{out} of the output of a variable node in $G(q_k)$ with connection degree i, is given by:

$$x_{out} = J_v(\sigma^2, (i-1)J_c^{-1}(x_{in}, q_k)\mathbf{1}_{q_k-1}, q_k) \ .$$

- The mutual information of extended message from $G(q_k)$ to $G(q_l)$ does not depend on which linear extension is used, but only on the group orders. Let x_{in} and x_{out} denote the mutual information of extension input and output, respectively. It follows from Definition 9

$$(1 - x_{out}) \log_2(q_l) = (1 - x_{in}) \log_2(q_k) \ .$$

- To express the mutual information of truncated message from $G(q_l)$ to $G(q_k)$, we use the LM-invariance property of input and output of the truncation operator. Let x_{in} and x_{out} denote the

mutual information of truncation input and output, respectively.

$$x_{out} = J_c(J_c^{-1}(x_{in}, q_l), q_k)$$

- Let $\mathbf{v}$ denote a probability vector, and $f(\mathbf{v})$ the corresponding Fourier Transform (FT) vector. Let $x_{\mathbf{v}}$ be the mutual information of a probability vector $\mathbf{v}$, and $x_{f(\mathbf{v})}$ denote the function given in equation (14) applied to the vector $f(\mathbf{v})$.

  Lemma 6: The connection between $x_{\mathbf{v}}$ and $x_{f(\mathbf{v})}$ is

  $$x_{f(\mathbf{v})} = 1 - x_{\mathbf{v}} \ .$$

  The proof is provided in Appendix. Through a check node in $G(q_l)$ with connection degree j, the mutual information transform from the FT perspective is equivalent to the one given by the reciprocal channel approximation [30]:

  $$x_{out} = 1 - J_c((j-1)J_c^{-1}(1 - x_{in}, q_l), q_l) \ .$$

  The reciprocal channel approximation used for hybrid LDPC codes is not looser than when it is used with non-binary LDPC codes, since the message densities are considered as, or projected on, Gaussian densities in both cases. However, by computer experiment, the approximation is looser than for binary LDPC codes in the first decoding iterations when the check node degree is very low ($j = 3$ or 4).

We obtain the whole extrinsic transfer function of one iteration of the hybrid LDPC decoder (equation (17)). The mutual information of a message going out of a check node of degree j in $G(q_l)$ at the t^{th} iteration and before truncation is denoted by $x_{cv}^{(j,l)^{(t)}}$. The same after truncation to become q_k sized is denoted $x_{cv,k}^{(j,l)^{(t)}}$. Analogously, the mutual information of a message going out of a variable node of degree i in $G(q_k)$ at the t^{th} iteration and before extension is denoted by $x_{vc}^{(i,k)^{(t)}}$. The same after extension to become q_l-sized is denoted $x_{vc,l}^{(i,k)^{(t)}}$.

$$x_{vc,l}^{(i,k)^{(t)}} = 1 - \frac{\log_2(q_k)}{\log_2(q_l)} \left(1 - x_{vc}^{(i,k)^{(t)}}\right) \tag{15}$$

$$x_{cv}^{(j,l)^{(t)}} = 1 - J_c \left((j-1) J_c^{-1} (1 - \sum_{i,k} \Pi(i,k|j,l) x_{vc,l}^{(i,k)^{(t)}}, q_l), q_l \right) \tag{16}$$

$$x_{cv,k}^{(j,l)^{(t)}} = J_c \left(J_c^{-1}(x_{cv}^{(j,l)^{(t)}}, q_l), q_k \right)$$

$$x_{vc}^{(i,k)^{(t+1)}} = J_v \left(\sigma^2, (i-1) J_c^{-1} (\sum_{j,l} \Pi(j,l|i,k) x_{cv,k}^{(j,l)^{(t)}}, q_k), q_k \right) \tag{17}$$

We also define the a posteriori (or cumulative) mutual information for each kind of variable node at the t^{th} iteration by

$$y^{(i,k)^{(t)}} = J_v\left(\sigma^2, i \cdot J_c^{-1}(\sum_{j,l}\Pi(j,l|i,k)x_{cv,k}^{(j,l)^{(t)}}, q_k), q_k\right) . \quad (18)$$

For any (i,k), $y^{(i,k)^{(t)}}$ is the quantity that must tend to 1 when t tends to infinity, for successful decoding. In the remainder, we refer to this mutual information evolution equation by using the notation $F(.)$ such that:

$$\{x_{vc}^{(i,k)^{(t+1)}}\}_{i,k} = F(\{x_{vc}^{(i,k)^{(t)}}\}_{i,k}, \Pi(i,j,k,l), \sigma^2) .$$

V. Distributions optimization

A. Context of the optimization

Let us denote the code rate R, and the target code rate R_{target}. The optimization procedure consists in finding $\Pi(i,j,k,l)$ which fulfills the following constraints at the lowest SNR:

$$\begin{aligned}
&\text{Code rate constraint:} && R = R_{target} \\
&\text{Sum constraint:} && \sum_{i,j,k,l}\Pi(i,j,k,l) = 1 \\
&\text{Sorting constraint:} && \Pi(i,j,k,l) = 0, \quad \forall(i,j,k,l) \text{ such that } q_k > q_l \quad (19) \\
&\text{Successful decoding condition:} && \lim_{t\to\infty} y^{(i,k)^{(t)}} = 1, \quad \forall(i,k) \quad (20) \\
& && \text{with } \{x_{vc}^{(i,k)^{(t+1)}}\}_{(i,k)} = F(\{x_{vc}^{(i,k)^{(t)}}\}_{(i,k)}, \Pi(i,j,k,l), \sigma^2)
\end{aligned}$$

B. Optimization with multi-dimensional EXIT charts

The successful decoding condition $\lim_{t\to\infty} y^{(i,k)^{(t)}} = 1$ for all (i,k), is verified by multi-dimensional EXIT charts. This technique, for hybrid LDPC codes, is a modification of the technique introduced in [31], and can be presented as follows:

1) Initialization: t=0. Set $x_{cv}^{(j,l)^{(0)}} = 0$ for all (j,l).
2) Compute $x_{vc}^{(i,k)^{(t)}}$ for all (i,k) with equation (17).
3) Compute $x_{cv}^{(j,l)^{(t)}}$ for all (j,l) with equation (16).
4) Compute $y^{(i,k)^{(t)}}$ for all (i,k) with equation (18).
5) If $y^{(i,k)^{(t)}} = 1$ up to the desired precision for all (i,k) then stop; otherwise $t = t+1$ and go to step 2.

Optimizing the detailed representation $\Pi(i,j,k,l)$, without any restriction on the parameters, requires to use multi-dimensional EXIT charts with a hill-climbing optimization method, like differential evolution. However, owing to the huge parameter space and multi-dimensional interpolations leading to a too high computational complexity, we restrict the parameter space to get a linear programming optimization problem.

C. Optimization with mono-dimensional EXIT charts

In this part, we consider the optimization of hybrid LDPC codes families with all check nodes in the same group $G(q_l)$ and with connection degrees independent of the variable nodes to which they are connected. We present how general equations (17) turns into mono-dimensional EXIT charts, and how this allows the use of linear programming for optimization. Let $x_e^{(t)}$ denote the averaged mutual information of extended messages. It is expressed in terms of the mutual information $x_{vc}^{(i,k)^{(t)}}$ of messages going out of variable nodes of degree i in $G(q_k)$, by simplification of equation (15):

$$x_e^{(t)} = 1 - \frac{1}{\log_2(q_l)} \sum_{i,k} \Pi(i,k) \log_2(q_k)(1 - x_{vc}^{(i,k)^{(t)}}) \ .$$

From equation (15), we can see that, for any (i,k,l):

$$\lim_{t\to\infty} x_{vc,l}^{(i,k)^{(t)}} = 1 \Leftrightarrow \lim_{t\to\infty} x_{vc}^{(i,k)^{(t)}} = 1$$

and then the successful decoding condition (20) reduces to

$$\lim_{t\to\infty} x_e^{(t)} = 1 \ .$$

By simplifying equation (17), $x_e^{(t+1)}$ can be expressed by a recursion in terms of $x_e^{(t)}$ as:

$$x_{cv,k}^{(j,l)^{(t)}} = J_c\left(J_c^{-1}\left(1 - J_c\left((j-1)J_c^{-1}(1-x_e^{(t)}, q_l), q_l\right), q_l\right), q_k\right) \ ;$$

$$x_e^{(t+1)} = \sum_{i,k} \Pi(i,k)\left(1 - \frac{log(q_k)}{log(q_l)}\left(1 - J_v\left(\sigma^2, (i-1)J_c^{-1}(\sum_j \Pi(j|i,k) x_{cv,k}^{(j,l)^{(t)}}, q_k)\mathbf{1}_{q_k-1}, q_k\right)\right)\right) \ . \tag{21}$$

Thus, the condition for successful decoding of hybrid LDPC codes in that specific case is

$$\forall t \geq 0, \quad x_e^{(t+1)} > x_e^{(t)} \tag{22}$$

In that case, the optimization procedure aims at finding distribution $\Pi(i,k|j,l)$ for given $\Pi(j,l)$. We see in equation (21) that $x_e^{(t+1)}$ depends linearly on $\Pi(i,k)$, turning the optimization problem into a linear programming problem.

VI. FINITE LENGTH OPTIMIZATION

This section presents an extension of optimization methods that has been described in [32] for finite-length non-binary LDPC codes with constant variable degree $d_v = 2$. We address the problem of the selection and the matching of the parity-check matrix $\mathbf{H}$ non-zero elements. In this section, we assume that the connectivity profile and group order profile of the graph have been optimized, with constant variable degree $d_v = 2$. With the knowledge of the graph connectivity, we run a PEG algorithm [33] in order to build a graph with a high girth.

The method is based on the binary image representation of $\mathbf{H}$ and of its components. First, the optimization of the rows of $\mathbf{H}$ is addressed to ensure good waterfall properties. Then, by taking into account the algebraic properties of closed topologies in the Tanner graph, such as cycles or their combinations, an iterative method is used to increase the minimum distance of the binary image of the code by avoiding low-weight codewords.

A. Row optimization

Based on the matrix representation of each non-zero element, we give hereafter the equivalent vector representation of the parity-check equations associated with the rows of $\mathbf{H}$.

Let $\mathbf{x} = [x_0 \dots x_{N-1}]$ be a codeword, and let p_j be the number of bits representing the binary map of symbol $x_j \in G(2^{p_j})$, $j = 0, \dots N-1$. For the i^{th} parity-check equation of H in the group $G(2^{p_i})$, we have the following vector equation:

$$\sum_{j:H_{ij}\neq 0} H_{ij}\mathbf{x_j} = \mathbf{0} \tag{23}$$

where H_{ij} is the $p_i \times p_j$ binary matrix representation of the non-zero element h_{ij}, $\mathbf{x_j}$ is the vector representation (binary map) of the symbol x_j. The all-zero component vector is denoted by $\mathbf{0}$.

Considering the i-th parity-check equation as a single component code, we define $\mathbf{H_i} = [H_{ij_0} \dots H_{ij_m} \dots H_{ij_{d_c-1}}]$ as its equivalent binary parity-check matrix, with $\{j_m : m = 0 \dots d_c - 1\}$ the indexes of the non-zero elements of the i-th parity-check equation. The size of $\mathbf{H_i}$ is $p_i \times (p_{ij_0} + \dots + p_{ij_{d_c-1}})$, with p_i and p_{ij_k} the extension orders of the groups of the check node and the k-th connected variable node, respectively. Let $\mathbf{X_i} = [\mathbf{x_{j_0}} \dots \mathbf{x_{j_{d_c-1}}}]^t$ be the binary representation of the symbols of the codeword $\mathbf{x}$ involved in the i^{th} parity-check equation. When using the binary representation, the i-th parity-check equation (23), can be written equivalently as $\mathbf{H_i}{\mathbf{X_i}}^t = \mathbf{0}^t$.

We define $d_{min}(i)$ as the minimum distance of the binary code associated with $\mathbf{H_i}$. As described in [32], a d_c-tuple of d_c linear maps is chosen in order to maximize the minimum distance $d_{min}(i)$ of

the code corresponding to the i^{th} row of $\mathbf{H}$, $i = 0, \ldots, M-1$. For hybrid LDPC codes, we adopt the same strategy, and choose for $\mathbf{H_i}$ a binary linear component code with the highest minimum distance achievable with the dimensions of $\mathbf{H_i}$ [34].

B. Avoiding low weight codewords

We now address the problem of designing codes with good minimum distance. It has been shown in [32] that the error floor of non-binary LDPC codes based on ultra-sparse ($d_v = 2$) graph is not uniquely due to pseudo-codewords, but also to low-weight codewords. Here we consider hybrid LDPC codes with constant variable degree $d_v = 2$. We adopt for hybrid LDPC codes the same strategy that has been introduced in [32], which aims at avoiding the low-weight codewords which are contained in the smallest cycles.

In order to do so, we first extract and store the cycles of the Tanner graph with length belonging to $\{g, \ldots, g + gap\}$, where g is the girth and gap is a small integer such that the number of cycles with size $g + gap$ is manageable. As in the previous section, we consider the binary images of cycles as component codes. Let $\mathbf{H}_{c_k}$ be the binary image of the k-th stored cycle. Since we consider $(2, d_c)$ codes, if some columns of $\mathbf{H}_{c_k}$ are linearly dependent, so will be the columns of $\mathbf{H}$. This means that a codeword of a cycle is also a codeword of the whole code. The proposed approach is hence to avoid low-weight codewords by properly choosing the linear maps implied in the cycles, so that no codeword of low weight is contained in the cycles. This is achieved by ensuring that the binary matrices corresponding to the cycles have full column rank. Hybrid LDPC codes are therefore well-suited to this kind of finite-length optimization procedure owing to the rectangular structure of the injective linear maps we consider as non-zero elements of the parity-check matrix.

VII. Numerical Results

A. Rate one-half codes

In the sequel, code rates are expressed in bits per channel use. We first give in Table I two code distributions and the corresponding thresholds for code rate one-half. Thresholds, denoted by $\left(\frac{E_b}{N_o}\right)^\star$, are approximated by Monte-Carlo simulations in the following manner. To mimic decoding of an infinite-length code, we consider a finite-length hybrid LDPC code corresponding to the given distribution. The length we used is 20000 coded bits. We send the all-zero codeword, and at each decoding iteration, the noise added to the codeword is changed, as well as the linear maps of the code, chosen uniformly at random. If the code is not structured, we also change the interleaver of the graph at each iteration. The

TABLE I

NODEWISE DISTRIBUTIONS OF THE HYBRID LDPC CODES USED FOR THE FINITE LENGTH SIMULATIONS.

	Hybrid LDPC code 1	Hybrid LDPC code 2
$\tilde{\Pi}(i=2, q_k=32)$		0.3950
$\tilde{\Pi}(i=2, q_k=64)$	0.4933	0.2050
$\tilde{\Pi}(i=2, q_k=256)$	0.4195	0.4000
$\tilde{\Pi}(i=6, q_k=64)$	0.0772	
$\tilde{\Pi}(i=6, q_k=256)$	0.0100	
$\tilde{\Pi}(j=5, q_l=256)$	0.5450	1
$\tilde{\Pi}(j=6, q_l=256)$	0.4550	
$\left(\frac{E_b}{N_o}\right)^\star$ (dB)	0.675	0.550

distribution is preserved because connection degrees and groups are not changed. An approximation of the threshold is the lowest SNR value for which the number of errors of the decoder reaches zero after 500 iterations. We then check that several approximations have small variance w.r.t. the average, and define the threshold as the average of the obtained approximations.

Thresholds are given only for the codes we are interested in for small codeword length applications. The channel is the BIAWGN channel with BPSK modulation. The hybrid LDPC code 1 is obtained by setting the different group orders, and then optimizing the connection profile of variable nodes for each group. We set the check node parameters (group order and connection profile), independently of the variable nodes parameters. Starting from $\Pi(i,j,k,l)$, the assumptions we consider on the parametrization, are translated into the following decomposition:

$$\begin{aligned}\Pi(i,j,k,l) &= \Pi(i,k,l)\Pi(j)\\ &= \Pi(i,k)\Pi(l|i,k)\Pi(j)\\ &= \Pi(i|k)\Pi(k)\Pi(l|k)\Pi(j)\end{aligned}$$

where $\Pi(i|k)\ \forall i$ is the connection profile of variable nodes in $G(q_k)$, which is optimized for all k. We fix $\Pi(k)$, $\Pi(l|k)$ and $\Pi(j)$. Check nodes have degree 5 or 6, independently of other parameters, and are in $G(256)$ as well as all redundancy variable nodes, while the information variable nodes are in $G(64)$. (Hence $\Pi(l|k)=\Pi(l)$). The connection profiles for these two groups are then optimized with maximum variable node degree equal to 10.

Remark: From a D-GLDPC codes perspective, variable nodes in the highest order group correspond to poor generalized component codes. It can be observed that the optimization procedure affects these nodes with as many high connection degrees as possible, given the constraints (i.e., when the code dimension K, equal to the log of the group order, is high, the length N is increased).

The hybrid LDPC code 2 is obtained by fixing the connection profile and optimizing the group orders of variable nodes. We set the check node parameters (group order and connection profile), independently of the variable nodes parameters. Starting from $\Pi(i,j,k,l)$, the assumptions we consider on the parametrization, are translated into the following decomposition:

$$\begin{aligned}\Pi(i,j,k,l) &= \Pi(i,k,l)\Pi(j)\\ &= \Pi(k|i,l)\Pi(i,l)\Pi(j)\\ &= \Pi(k|i,l)\Pi(i)\Pi(l)\Pi(j)\end{aligned}$$

We aim at optimizing as many group order profiles $\Pi(k|i,l)$, $\forall k$, as the number of different variable node connection degrees i. We fix $\Pi(i)$, $\Pi(l)$ and $\Pi(j)$. The graph connections are set regular with constant variable degree $d_v = 2$ and constant check degree $d_c = 5$. All check nodes are fixed to be in $G(256)$. Therefore $\Pi(k|i,l) = \Pi(k|i)$. Hybrid LDPC code 1 and 2 ensembles are hence unstructured code ensembles.

From Table I, we can say that hybrid LDPC codes do not outperform non-binary LDPC codes in terms of decoding thresholds. Moreover, both kinds of codes can have better thresholds than those in Table I by allowing higher connection degrees. However, our purpose is to point out the good finite length performance of hybrid LDPC codes, that can be evaluated by the error-floor behavior, as explained in Section I. Since the error-floor is due to cycles in the graph, best error-floor performance are usually reached with low connection degrees, i.e, with sparser graphs [25]. That is why we have focused on low connection degrees. For such low degrees, hybrid LDPC codes do not approach the capacity as close as multi-edge type LDPC codes do, but their thresholds are in the range of protograph-based LDPC code thresholds [35][31]. This is due to the adopted detailed representation Π which cannot handle degree one variable nodes. However, it would be an interesting perspective to switch from the detailed representation to a multi-edge type representation for hybrid LDPC codes. This will certainly enable to get capacity-approaching distributions with low connection degrees. Indeed, it has been shown in [15] that introducing degree-1 variable nodes in non-binary LDPC codes makes the decoding threshold getting closer to the theoretical limit.

In the sequel, for all simulated hybrid LDPC codes except hybrid LDPC code 1 in Figure 3, the linear

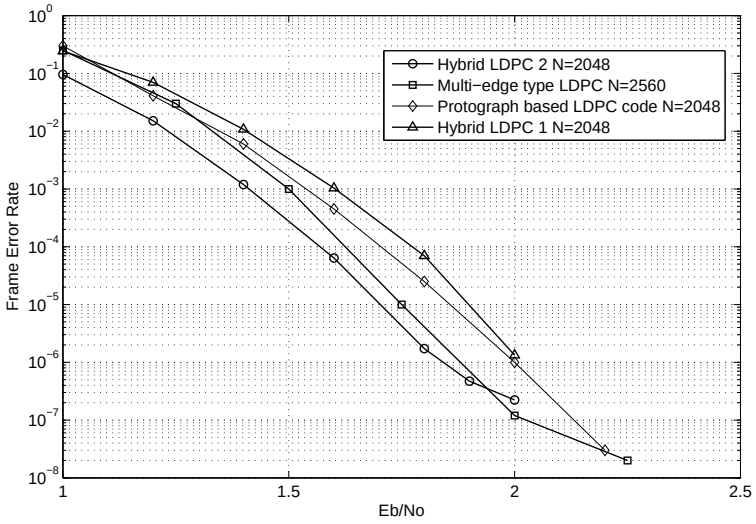

Fig. 3. FER versus $\frac{E_b}{N_o}$ (in dB): code rate one-half. $N_{bit} = 2048$ coded bits except for the multi-edge type LDPC code for which $N_{bit} = 2560$ coded bits. $N_{iter} = 500$ decoding iterations are performed.

maps are chosen according to the technique presented in Section VI. Detailed simulation results (numbers of frames in error and simulated) are presented in Appendix. Figure 3 represents frame error rate (FER) curves for different codes with code rate one-half. The performance curves of hybrid LDPC codes 1 and 2 are compared with a protograph-based LDPC code from [35], and a multi-edge type (MET) LDPC code from [25]. This code has been specifically designed for low error-floor. All codes have $N_{bit} = 2048$ coded bits, except the MET LDPC code which has $N_{bit} = 2560$ coded bits.

The graphs of hybrid LDPC codes have been built with the random PEG algorithm described in [36]. To create systematic hybrid LDPC codes with the method of [36], the modification of the technique in [36] is the same as what is described in [33] (Section V) to create upper-triangular encoding matrices for LDPC codes. It is worth noting that the input of the graph construction method is only the connection profile of the code without the group order profile.

We see that the hybrid LDPC code 1 has performance very close to the protograph-based LDPC code in the simulated range of SNR. The hybrid LDPC code 2 has slightly better waterfall and slightly higher error-floor than the MET LDPC code which is about 500 bits longer. Hybrid LDPC codes are therefore capable of exhibiting performance equivalent to MET LDPC codes, which are, to the best of our knowledge, among the most interesting structured codes. It is worth noting that, unlike MET and

protograph-based LDPC codes, the presented hybrid LDPC codes are non-structured codes.

Furthermore, for sake of clarity in the figures, we did not plot irregular $GF(256)$ LDPC codes performance. However, we can mention that such codes cannot outperform regular $(2, 4)$ $GF(256)$ and hybrid LDPC codes, even not optimized for finite-length (Section VI), in terms of error-floor. Thus, even though they can provide better waterfall performance than regular $(2, 4)$ $GF(256)$ or hybrid LDPC codes which have lower connection degrees can, irregular $GF(256)$ LDPC codes do not allow to lower the error-floor as much as with connection-regular codes. Hence irregular $GF(256)$ LDPC codes do not allow to get the same amplitude in the choice of the tradeoff between waterfall and error-floor performance, as hybrid LDPC codes do.

B. Rate one-sixth codes

For communication systems operating in the low SNR regime (e.g., code-spread communication systems and power-limited sensor networks), low-rate coding schemes play a critical role. Although LDPC codes can exhibit capacity-approaching performance for various code rates when the ensemble profiles are optimized [4], in the low-rate region, it is difficult to obtain good low-rate LDPC codes. The analytical reason for that is given in [37] (Section II.D): "lower rate LDPC codes require larger SNR increase from the decoding threshold to obtain similar conditions regarding decoding tunnels in EXIT charts than their higher rate counterparts". We intend to illustrate the interest of hybrid LDPC codes for low-rate applications requiring short block length (from 200 to 1000 information bits).

TABLE II

NODEWISE DISTRIBUTION OF THE RATE $\frac{1}{6}$ HYBRID LDPC CODE

	Hybrid LDPC code $R = \frac{1}{6}$
$\tilde{\Pi}(i = 2, q_k = 8)$	0.227
$\tilde{\Pi}(i = 2, q_k = 16)$	0.106
$\tilde{\Pi}(i = 2, q_k = 256)$	0.667
$\tilde{\Pi}(j = 3, q_l = 256)$	1
$\left(\frac{E_b}{N_o}\right)^{\star}$ (dB)	-0.41
Capacity (dB)	-1.07

In Figure 4, Bit Error Rates (BER) of a rate $1/6$ hybrid LDPC code, whose distribution is given in Table II, are compared with Turbo Hadamard code (TH) taken from [38] and Zigzag Hadamard (ZH) code taken

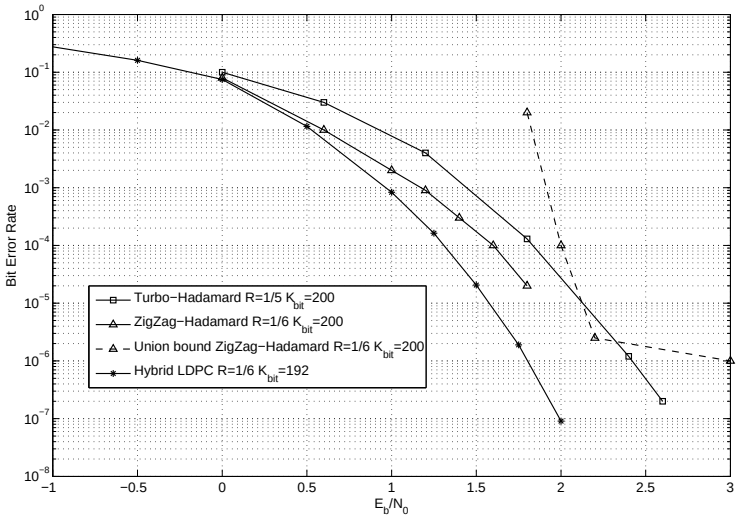

Fig. 4. Comparison of hybrid LDPC code with Turbo Hadamard code (TH) taken from [38] and Zigzag Hadamard (ZH) code taken from [39], for $K_{bit} \simeq 200$ information bits. The number of decoding iterations is $N_{iter} = 30$ for Turbo Hadamard codes, and $N_{iter} = 200$ for the hybrid LDPC code.

from [39], for $K_{bit} \simeq 200$ information bits. The number of decoding iterations is $N_{iter} = 30$ for Turbo Hadamard codes, and $N_{iter} = 200$ for the hybrid LDPC code. However, the comparison is not unfair because the number of iterations for Turbo-like codes and that for LDPC codes does not scale identically with performance as, e.g., pointed out in [40]. This can be interpreted by the fact that the complexity per iteration of Turbo-like codes is higher than that of LDPC codes, owing to the BCJR algorithm run at each iteration. The hybrid LDPC code outperforms with 0.3 dB gain the ZH code. Additionally, the hybrid code has no observed error floor up to a BER=10^{-7}. When comparing the computer simulation of the hybrid LDPC code with the union bound of ZH code, we observe that the BER of the hybrid LDPC code has gain of about one decade at $\frac{E_b}{N_0} = 2$dB. This gives a hint to predict that the error floor of the hybrid LDPC code is lower than the error floor of the ZH code.

In Figure 5, the FER comparison is drawn for code rate $1/6$ and $K_{bit} \simeq 1000$ information bits. The quasi-cyclic LDPC code is designed to have low error-floor [42]. The hybrid LDPC code is better than the quasi-cyclic LDPC in the waterfall region. However, the error-floor of the quasi-cyclic LDPC code is not provided in [42], and we were not able to evaluate the minimum distance of the hybrid LDPC code. Indeed, unlike quasi-cyclic LDPC codes, the proposed hybrid LDPC code is not structured. For unstructured LDPC codes, the minimum distance can be evaluated by the method presented in [43]. In

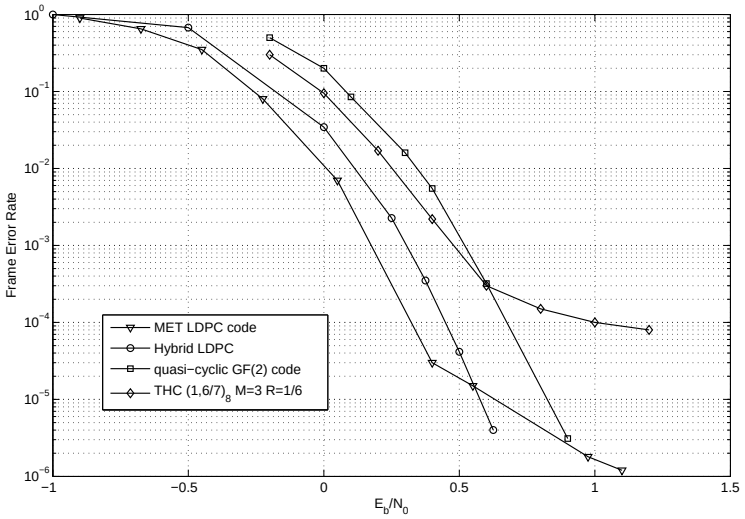

Fig. 5. Comparison of hybrid LDPC code with punctured Turbo Hadamard (PTH) taken from [41] and other powerful codes, for code rate $1/6$. The PTH code has $K_{bit} = 999$ information bits, and the other codes have $K_{bit} = 1024$ information bits. $N_{iter} = 50$ for the PTH code, and $N_{iter} = 200$ for the other codes.

Figure 5, the codeword length is 6144 bits, which results in a too much high complexity to implement the technique of [43]. That is why we were unable to approximate the minimum distance of the hybrid LDPC code for this codeword length. The hybrid LDPC code is better than the PTH codes both in the waterfall and in the error-floor regions. The hybrid LDPC code has poorer waterfall region than the MET LDPC code [44], but better error-floor. Hence, for rate $1/6$ too, the performance of hybrid LDPC codes are equivalent to the one of MET LDPC codes, by allowing to reach comparable trade-off between waterfall and error-floor performance.

Remark: Let us mention that hybrid LDPC codes, with injective linear maps as non-zero elements, are well-fitted to low code rates thanks to their structure. Indeed, like all other kinds of codes with generalized constraint nodes (Turbo Hadamard code [38], LDPC Hadamard codes [45], GLDPC [13], D-GLDPC [14], or Tail-biting LDPC [15]), they are well-fitted to low code rates because the graph rate is higher than the code rate. This can help the iterative decoding: when the code rate is very low, decoding on a higher rate graph can lead to better performance.

VIII. Conclusions

A new class of LDPC codes, named hybrid LDPC codes, has been introduced. Asymptotic analysis of this class of codes has been carried out for distribution optimization, as well as finite-length optimization. Numerical simulations, for code rates one-half and one-sixth, illustrate that hybrid LDPC codes can be good competitors for the best known codes, like protograph-based or MET LDPC codes, by allowing to reach interesting trade-off between waterfall and error-floor performances.

Appendix

Lemma 1 *Let $P_e^{(t)}(\mathbf{x})$ denote the conditional error probability after the t^{th} BP decoding iteration of a $GF(q)$ LDPC code, assuming that codeword $\mathbf{x}$ was sent. If the channel is symmetric, then $P_e^{(t)}(\mathbf{x})$ is independent of $\mathbf{x}$.*

Proof: The proof has the same structure as the proof of Lemma 1 in [5]. The notations are the same as in [5] and Section III-F.

Let $\Psi_v^{(t)}(\mathbf{m}_0, \mathbf{m}_1, \ldots, \mathbf{m}_{d_v-1})$ denote the message map of any variable node at iteration t, according to equation (7). The size of argument messages is implicitly the one of the group of the variable node. Let $\Psi_c^{(t)}(\mathbf{m}_1, \ldots, \mathbf{m}_{d_c-1})$ be the message map of any check node. The sizes of argument messages are implicitly the one of the group of each variable node connected to the check node, according to equation (6).

- Check node symmetry: Let G be the Cartesian product group defined in Section III-F. For any sequence $(b_1, \ldots, b_{d_c-1})$ in G such that $\bigoplus_{i=1}^{d_c-1} A_{v_i c} b_i \in Im(A_{vc})$, we have (see equation (6))

$$\Psi_c^{(t)}(\mathbf{m}_1^{+b_1}, \ldots, \mathbf{m}_{d_c-1}^{+b_{d_c-1}}) = \Psi_c^{(t)}(\mathbf{m}_1, \ldots, \mathbf{m}_{d_c-1})^{+A_{vc}^{-1}(\bigoplus_{i=1}^{d_c-1} A_{v_i c} b_i)}$$

- Variable node symmetry: We also have, for any $b \in GF(q_v)$:

$$\Psi_v^{(t)}(\mathbf{m}_0^{+b}, \mathbf{m}_1^{+b}, \ldots, \mathbf{m}_{d_v-1}^{+b}) = \Psi_v^{(t)}(\mathbf{m}_1, \ldots, \mathbf{m}_{d_c-1})^{+b}$$

Let $\mathbf{Z}_i$ denote the random variable being the channel output in probability form, conditionally to the transmission of the zero symbol. Each $\mathbf{Z}_i$ for any $i = 1 \ldots N$ has same size as the group of the corresponding codeword symbol. Any memoryless symmetric channel can be modeled as

$$\mathbf{Y}_i = \mathbf{Z}_i^{+x_i}$$

where x_i is the i^{th} component of $\mathbf{x}$ which is a vector of size N, denoting an arbitrary codeword of the hybrid LDPC code. The channel output in probability form $\mathbf{Y}_i$ results from the transmission of $\mathbf{x}$.

Let v denote an arbitrary variable node and let c denote one of its neighboring check nodes. For any observation in probability form $\mathbf{w}$, let $\mathbf{m}_{vc}^{(t)}(\mathbf{w})$ denote the message sent from v to c in iteration t assuming $\mathbf{w}$ was received. The quantity $\mathbf{w}$ is hence a set of channel output vectors in probability form $\mathbf{w}_i$, for all $i = 1 \dots N$. The same definition holds for $\mathbf{m}_{cv}^{(t)}(\mathbf{w})$ from c to v. From the variable node symmetry at $t = 0$ we have $\mathbf{m}_{vc}^{(0)}(\mathbf{y}) = \mathbf{m}_{vc}^{(0)}(\mathbf{z})^{+x_v}$. Assuming now that in iteration t we have $\mathbf{m}_{vc}^{(t)}(\mathbf{y}) = \mathbf{m}_{vc}^{(t)}(\mathbf{z})^{+x_v}$. Since $\mathbf{x}$ is a codeword, we have $\bigoplus_{i=1}^{d_c} A_{v_i c} x_i = 0$, and hence $\bigoplus_{i=1}^{d_c - 1} A_{v_i c} x_i = A_{vc} x_v$. From the check node symmetry condition we conclude that

$$\mathbf{m}_{cv}^{(t+1)}(\mathbf{y}) = \mathbf{m}_{cv}^{(t+1)}(\mathbf{z})^{+x_v} \ .$$

Moreover, from the variable node symmetry condition, it follows that in iteration $t+1$ the message sent from v to c is

$$\mathbf{m}_{vc}^{(t+1)}(\mathbf{y}) = \mathbf{m}_{vc}^{(t+1)}(\mathbf{z})^{+x_v} \ .$$

Thus, all messages to and from variable node v when $\mathbf{y}$ is received are permutations by x_v of the corresponding message when $\mathbf{z}$ is received. Hence, both decoders commit exactly the same number of errors, which proves the lemma.

□

Lemma 2 *If the channel is symmetric, then, under the all-zero codeword assumption, the density P_0 of the initial message in LDR form is symmetric:*

$$P_0(\mathbf{W} = \mathbf{w}) = e^{w_i} P_0(\mathbf{W} = \mathbf{w}^{+i})$$

Proof: Let x_{noisy} be the noisy observation of the sent symbol x. Let $L(.)$ be the surjective map which relates the noisy observation to a LDR vector: $\mathbf{W} = L(x_{\text{noisy}})$. The set of observations resulting in LDR vector $\mathbf{w}$ is denoted by $L^{-1}(\mathbf{w})$. Thus we have, for all $i \in G(q)$, $P(\mathbf{W} = \mathbf{w}|x = i) = P(x_{\text{noisy}} \in L^{-1}(\mathbf{w})|x = i)$.

Furthermore, let $\mathbf{y}$ be $\mathbf{y} = LDR^{-1}(\mathbf{w})$ and $\mathbf{Y}$ be $\mathbf{Y} = LDR^{-1}(\mathbf{W})$. By definition of Y_i, for all $i \in G(q)$, $Y_i = P(x_{\text{noisy}}|x = i)$, therefore we also have $Y_i = P(x_{\text{noisy}} \in L^{-1}(\mathbf{W})|x = i)$ and $y_i = P(x_{\text{noisy}} \in L^{-1}(\mathbf{w})|x = i)$. Owing to the channel symmetry, we have $P(\mathbf{Y} = \mathbf{y}^{+i}|x = 0) = P(\mathbf{Y} = \mathbf{y}|x = i)$.

Let us prove that $P_0(\mathbf{W}=\mathbf{w})$ satisfies equation (10):

$$\begin{aligned}
e^{w_i}P_0(\mathbf{W}=\mathbf{w}^{+i}) &= e^{w_i}P(\mathbf{W}=\mathbf{w}^{+i}|x=0) \\
&= \frac{P(x_{\text{noisy}}\in L^{-1}(\mathbf{w})|x=0)}{P(x_{\text{noisy}}\in L^{-1}(\mathbf{w})|x=i)}P(\mathbf{Y}=\mathbf{y}^{+i}|x=0) \\
&= \frac{P(x_{\text{noisy}}\in L^{-1}(\mathbf{w})|x=0)}{P(x_{\text{noisy}}\in L^{-1}(\mathbf{w})|x=i)}P(\mathbf{Y}=\mathbf{y}|x=i) \\
&= \frac{P(x_{\text{noisy}}\in L^{-1}(\mathbf{w})|x=0)}{P(x_{\text{noisy}}\in L^{-1}(\mathbf{w})|x=i)}P(\mathbf{W}=\mathbf{w}|x=i) \\
&= \frac{P(x_{\text{noisy}}\in L^{-1}(\mathbf{w})|x=0)}{P(x_{\text{noisy}}\in L^{-1}(\mathbf{w})|x=i)}P(x_{\text{noisy}}\in L^{-1}(\mathbf{w})|x=i) \\
&= P(x_{\text{noisy}}\in L^{-1}(\mathbf{w})|x=0) \\
&= P(\mathbf{W}=\mathbf{w}|x=0) \\
&= P_0(\mathbf{W}=\mathbf{w})
\end{aligned}$$

□

Lemma 3 *If the bipartite graph is cycle-free, then, under the all-zero codeword assumption, all the messages on the graph at any iteration, are symmetric.*

Proof: When hybrid LDPC codes are decoded with BP, both data pass and check pass steps are the same as classical non-binary codes decoding steps. Since these steps preserve symmetry [4] if the graph is cycle-free, the following Lemma 7 ensures that the hybrid decoder preserves the symmetry property if the input messages from the channel are symmetric.

Lemma 7: If $\mathbf{X}$ and $\mathbf{Y}$ are a symmetric LDR random vectors, then the extension $\mathbf{X}^{\times A}$ of $\mathbf{X}$, by any full-rank linear extension A, remains symmetric. The same for the truncation $\mathbf{Y}^{\times A^{-1}}$ of $\mathbf{Y}$ by the inverse of A.

Proof: We first prove that any q_2-sized extension of a q_1-sized symmetric random vector remains symmetric. We want to show that

$$\forall b\in\{0,\ldots,q_2-1\}, P(\mathbf{X}^{\times A}=\mathbf{y}) = e^{y_b}P(\mathbf{X}^{\times A}=\mathbf{y}^{+b})\ . \tag{24}$$

Case $b\notin Im(A)$:

- In the case where $y_b\neq-\infty$:

We have to show that

$$e^{-y_b} P(\mathbf{X}^{\times A} = \mathbf{y}) = P(\mathbf{X}^{\times A} = \mathbf{y}^{+b}) \ .$$

If $y_b \neq \infty$, then $P(\mathbf{X}^{\times A} = \mathbf{y}) = 0$. If $y_b = \infty$, then $e^{-y_b} = 0$. Thus, we have to show that

$$\forall b \notin Im(A), P(\mathbf{X}^{\times A} = \mathbf{y}^{+b}) = 0 \ . \tag{25}$$

To prove equation (25), it is sufficient to show that $\exists i \notin Im(A)$ such that $y_i^{+b} \neq \infty$. We have $y_i^{+b} = y_{b+i} - y_b$. It is sufficient to choose $i = b$, then $y_b^{+b} = -y_b$. Since $y_b^{+b} = -y_b \neq \infty$ by hypothesis, $P(\mathbf{X}^{\times A} = \mathbf{y}^{+b}) = 0$.

- In the case $y_b = -\infty$, to prove equation (24), we have to prove that $P(\mathbf{X}^{\times A} = \mathbf{y}) = 0$, which is straightforward because $b \notin Im(A)$, and hence $P(\mathbf{X}^{\times A} = \mathbf{y}) \neq 0 \Rightarrow y_b = \infty$. By taking the contraposition, we end up with the sought result.

Hence we have proved equation (24) in the case of $b \notin Im(A)$.

Case $b \in Im(A)$:

Let $\delta_{x,y}$ be the Kronecker delta function whose value is 1 if $x = y$, 0 otherwise. We have

$$P(\mathbf{X}^{\times A} = \mathbf{y}) = P(\mathbf{X} = \mathbf{y}^{\times A^{-1}}) \prod_{i \notin \mathrm{Im}(A)} \delta_{y_i,\infty} \ .$$

Since b belongs to $Im(A)$, we denote by a the element in $\{0, \ldots, q_1 - 1\}$ such that $b = Aa$. The input message $\mathbf{X}$ is symmetric, hence we have

$$P(\mathbf{X} = \mathbf{y}^{\times A^{-1}}) = e^{y_{Aa}} P(\mathbf{X} = (\mathbf{y}^{\times A^{-1}})^{+a})$$

Recall that, for any extension A, we have $y_a^{\times A^{-1}} = y_{Aa}$.

$$\begin{aligned} \forall i \in \{0, \ldots, q_1 - 1\}, \quad (\mathbf{y}^{\times A^{-1}})_i^{+a} &= y_{i+a}^{\times A^{-1}} - y_a^{\times A^{-1}} \\ &= y_{A(i+a)} - y_{Aa} \\ &= y_{Ai}^{+Aa} \\ &= (y^{+Aa})_i^{\times A^{-1}} \end{aligned}$$

Thus

$$P(\mathbf{X}^{\times A} = \mathbf{y}) = e^{y_{Aa}} P(\mathbf{X} = (\mathbf{y}^{+Aa})^{\times A^{-1}}) \prod_{i \notin \mathrm{Im}(A)} \delta_{y_i,\infty} \ . \tag{26}$$

We note that:

$$P(\mathbf{X}^{\times A} = \mathbf{y}^{+Aa}) = P(\mathbf{X} = (y^{+Aa})^{\times A^{-1}}) \prod_{j \notin \mathrm{Im}(A)} \delta_{y_j^{+Aa},\infty} \ . \tag{27}$$

In this case, for all $j \in \{0,\dots,q_2-1\}$, $\delta_{y_j^{+Aa},\infty} = \delta_{y_{Aa+j}-y_{Aa},\infty} = \delta_{y_{Aa+j},\infty}$. For all $i \in \{0,\dots,q_2-1\}$, if $i \notin Im(A)$, then $\exists j \notin Im(A)$: $i = Aa + j$. Therefore $\{i \in \{0,\dots,q_2-1\} \text{ s.t. } i \notin Im(A)\} = \{j \in \{0,\dots,q_2-1\} \text{ s.t. } Aa+j \notin Im(A)\}$. We finally obtain:

$$\prod_{j \notin \text{Im}(A)} \delta_{y_j^{+Aa},\infty} = \prod_{i \notin \text{Im}(A)} \delta_{y_i,\infty} \ .$$

The above equality allows to insert equation (27) into equation (26). We can now conclude that, when b is in $Im(A)$, equation (24) is satisfied.

This completes the proof of the first part of Lemma 7.

We now prove that any truncation of a symmetric random LDR vector remains symmetric. We have to prove that

$$\forall a \in \{0,\dots,q_1-1\}, \quad P(\mathbf{Y}^{\times A^{-1}} = \mathbf{x}) = e^{x_a} P(\mathbf{Y}^{\times A^{-1}} = \mathbf{x}^{+a}) \ . \tag{28}$$

Let b be the image of a by A: $b = Aa$.

$$\begin{aligned} P(\mathbf{Y}^{\times A^{-1}} = \mathbf{x}) &= \sum_{\substack{\mathbf{y}: y_0 = x_0, \\ y_{A1} = x_1, \\ \dots, \\ y_{A(q_1-1)} = x_{q_1-1}}} P(\mathbf{Y} = \mathbf{y}) \\ &= \sum_{\substack{\mathbf{y}: y_0 = x_0, \\ y_{A1} = x_1, \\ \dots, \\ y_{A(q_1-1)} = x_{q_1-1}}} e^{y_b} P(\mathbf{Y} = \mathbf{y}^{+b}) \\ &= e^{x_a} \sum_{\substack{\mathbf{y}: y_0 = x_0, \\ y_{A1} = x_1, \\ \dots, \\ y_{A(q_1-1)} = x_{q_1-1}}} P(\mathbf{Y} = \mathbf{y}^{+b}) \end{aligned}$$

We note that:

$$\forall i \in Im(A), \quad y_i^{+Aa} = y_{Aa+i} - y_{Aa} = x_{a+A^{-1}i} - x_a = (x^{+a})_i^{\times A} \ ,$$

where the last step is infered thanks to equation (2). Thus

$$\begin{aligned} P(\mathbf{Y}^{\times A^{-1}} = \mathbf{x}) &= e^{x_a} \sum_{\substack{\mathbf{y}: y_0 = (x^{+a})_0^{\times A}, \\ y_{A1} = (x^{+a})_{A1}^{\times A}, \\ \dots, \\ y_{A(q_1-1)} = (x^{+a})_{A(q_1-1)}^{\times A}}} P(\mathbf{Y} = \mathbf{y}) \\ &= e^{x_a} P(\mathbf{Y}^{\times A^{-1}} = \mathbf{x}^{+a}) \end{aligned} \tag{29}$$

We have obtained equation (28).

■

This completes the proof of Lemma 3.

□

Lemma 8: $E_{k,l}$ denotes the set of extensions from $G(q_k)$ to $G(q_l)$. For given k and l,

$$\forall (i,j) \in \{1,\ldots,q_k-1\} \times \{1,\ldots,q_l-1\}, \quad \frac{\text{Card}(A \in E_{k,l} : A^{-1}j = i)}{\text{Card}(E_{k,l})} = \frac{1}{q_l - 1}$$

Proof: p_k and p_l denote $\log_2(q_k)$ and $\log_2(q_l)$, respectively.

Without any constraint to build a linear extension A from $G(q_k)$ to $G(q_l)$, except the one of full-rank, we have $2^{p_l} - 2^{n-1}$ choices for the n^{th} row, $n = 1,\ldots,p_l$.

For given i and j, with the constraint that $Ai = j$, we have $2^{p_l - b_i} + 2^{\lfloor \frac{b_i}{2} \rfloor} - 2^{n-1}$ choices for the n^{th} row, $n = 1,\ldots,p_l$, where b_i is the number of bits equal to 1 in the binary map of α_i. Thus, the number of A such that $Ai = j$ is dependent only on i. Let say

$$\text{Card}(A \in E_{k,l} : A^{-1}j = i) = \beta_i$$

we have

$$\sum_{j=1}^{q_l-1} \text{Card}(A \in E_{k,l} : Ai = j) = \text{Card}(E_{k,l})$$

Therefore

$$\forall (i,j) \in \{1,\ldots,q_k-1\} \times \{1,\ldots,q_l-1\}, \qquad \frac{\text{Card}(A \in E_{k,l} : Ai = j)}{\text{Card}(E_{k,l})} = \frac{1}{q_l - 1}$$

Lemma 4. *If a random probability-vector* $\mathbf{Y}$ *of size* q_l *is LM-invariant, then for all* $(m,n) \in \{0,\ldots,q_l-1\} \times \{0,\ldots,q_l-1\}$*, the random variables* Y_m *and* Y_n *are identically distributed.*

Proof: For any (q_k, q_l), $q_k < q_l$, $T_{l,k}$ denotes the set of all truncations from $G(q_l)$ to $G(q_k)$. We assume $\mathbf{Y}$ LM-invariant. A^{-1} and B^{-1} denote two truncations independently arbitrary chosen in $T_{l,k}$. For any m and n in $\{0,\ldots,q_l-1\}$, we can choose extension A such that $m \in$ Im(A) and $A^{-1}m$ is denoted by i. Also, we choose B such that $Bi = n$. $\mathbf{Y}$ LM-invariant implies

$$\forall (i, A^{-1}, B^{-1}) \in \{0,\ldots,q_k-1\} \times T_{l,k} \times T_{l,k}, P(Y_i^{\times A^{-1}} = x) = P(Y_i^{\times B^{-1}} = x)$$

This is equivalent to

$$P(Y_{Ai} = x) = P(Y_{Bi} = x)$$

and hence

$$P(Y_m = x) = P(Y_n = x), \qquad \forall (m,n) \in \{0,\dots,q_l-1\} \times \{0,\dots,q_l-1\}$$

□

Lemma 5. *Consider a random vector* $\mathbf{Y}$ *of size* q_l*. If there exist* q_k *and a random vector* $\mathbf{X}$ *of size* q_k *such that* $\mathbf{Y} = \tilde{\mathbf{X}}$*, then* $\mathbf{Y}$ *is LM-invariant.*

Proof: We want to prove that, for all $n < l$, for any $(B,C) \in E_{n,l} \times E_{n,l}$, $Y^{\times B^{-1}}$ and $Y^{\times C^{-1}}$ are identically distributed.

By hypothesis $\mathbf{Y} = \mathbf{X}^{\times A}$, with $\mathbf{X}$ of size q_k and A uniformly chosen at random in $E_{k,l}$. Let $D^{(B)}$ be $D^{(B)} = B^{-1} \circ A$ and $D^{(C)}$ be $D^{(C)} = C^{-1} \circ A$, where $\circ$ stands for the composition of functions. Then $Y^{\times B^{-1}} = X^{\times D^{(B)}}$ and $Y^{\times C^{-1}} = X^{\times D^{(C)}}$. Let $\mathbf{v}^{(B)}$ be a random vector of size q_n defined by:

$$\forall i = 0,\dots,q_n-1, \quad v_i^{(B)} = \begin{cases} A^{-1}(Bi) & \text{if } Bi \in Im(A); \\ 0 & \text{otherwise.} \end{cases}$$

We can define the random vector $\mathbf{v}^{(C)}$ in a similar way. With such definitions, when $\mathbf{X}$ is a probability vector, we have:

$$\forall i = 0,\dots,q_n-1, \quad Y_i^{\times B^{-1}} = \begin{cases} X_{v_i^{(B)}} & \text{if } v_i^{(B)} \neq 0; \\ 0 & \text{otherwise.} \end{cases} \tag{30}$$

(The same holds when $\mathbf{X}$ is a LDR vector by replacing 0 by ∞.) We end up with the sought result by showing that $\mathbf{v}^{(B)}$ and $\mathbf{v}^{(C)}$ are identically distributed (we recall that B and C are fixed while A is chosen uniformly at random). For all $\mathbf{m}$ of size q_n in $G(q_k)^{q_n}$, we define the events E and F:

- the event E that for all p such that $m_p \neq 0$, A is such that $Am_p = Bp$,
- the event F that for all p such that $m_p = 0$, A is such that: there is no $i \in G(q_k)$ such that $Ai = Bp$.

Thus we have

$$P(\mathbf{v}^{(B)} = \mathbf{m}) = P(E \cap F)\ .$$

In the proof of Lemma 8, we have proven that, for all given $i \in G(q_k)$ and $j \in G(q_l)$, P('A is such that Ai=j') is dependent only on i. Thus, it is easy to see that $P(\mathbf{v}^{(B)} = \mathbf{m})$ does not depend on B. $\mathbf{v}^{(B)}$ and $\mathbf{v}^{(c)}$ are therefore identically distributed, so are $Y^{\times B^{-1}}$ and $Y^{\times C^{-1}}$ owing to equation (30). This completes the proof.

□

Lemma 9: The product of two LM-invariant random vectors is LM-invariant.

Proof: Let $\mathbf{U}$ and $\mathbf{V}$ be two LM-invariant random vectors of size q_l. For any $q_k < q_l$, let A and B be any two linear maps from $G(q_k)$ to $G(q_l)$. Since $\mathbf{U}$ is LM-invariant, $\mathbf{U}^{\times A^{-1}}$ and $\mathbf{U}^{\times B^{-1}}$ are identically distributed, by definition of LM-invariance. The same holds for $\mathbf{V}$. $\mathbf{U}^{\times A^{-1}}\mathbf{V}^{\times A^{-1}}$ and $\mathbf{U}^{\times B^{-1}}\mathbf{V}^{\times B^{-1}}$ are therefore identically distributed. Moreover, it is clear that $\mathbf{U}^{\times A^{-1}}\mathbf{V}^{\times A^{-1}} = (\mathbf{UV})^{\times A^{-1}}$, for any A. Hence, $(\mathbf{UV})^{\times A^{-1}}$ and $(\mathbf{UV})^{\times B^{-1}}$ is LM-invariant. This completes the proof.

□

Proof of Theorem 1: $\mathbf{X}^{(k)}$ denotes a random probability-vector of size q_k. The j^{th} component of the random truncation of $\mathbf{X}^{(k)}$ is denoted by $\frac{\mathrm{rt}}{X_j^{(k)}}$. The j^{th} component of the random extension of $\mathbf{X}^{(k)}$ is denoted by $\frac{\mathrm{re}}{X_j^{(k)}}$. The j^{th} component of the random extension followed by a random truncation of $\mathbf{X}^{(k)}$ is denoted by $\frac{\mathrm{rt+re}}{X_j^{(k)}}$.

We define the operator D_a by:

$$D_a(\mathbf{X}^{(l)}) \quad = \quad \frac{1}{q_l - 1} \sum_{j=1}^{q_l - 1} \mathbb{E}\left(\sqrt{\frac{X_j^{(l)}}{X_0^{(l)}}} \right) .$$

The following equalities are hence deduced from the previous definitions:

$$\begin{aligned}
\mathbb{E}\left(\sqrt{\frac{\frac{\mathrm{re}}{X_j^{(k)}}}{\frac{\mathrm{re}}{X_0^{(k)}}}} \right) &= \sum_l \Pi(l|k) \frac{1}{q_l - 1} \sum_{i=1}^{q_k - 1} \mathbb{E}\left(\sqrt{\frac{X_i^{(k)}}{X_0^{(k)}}} \right) \\
\mathbb{E}\left(\sqrt{\frac{\frac{\mathrm{rt}}{X_i^{(l)}}}{\frac{\mathrm{rt}}{X_0^{(l)}}}} \right) &= \frac{1}{q_l - 1} \sum_{j=1}^{q_l - 1} \mathbb{E}\left(\sqrt{\frac{X_j^{(l)}}{X_0^{(l)}}} \right) \\
&= D_a(\mathbf{X}^{(l)}) \\
\mathbb{E}\left(\sqrt{\frac{\frac{\mathrm{rt+re}}{X_i^{(k)}}}{\frac{\mathrm{rt+re}}{X_0^{(k)}}}} \right) &= \sum_l \Pi(l|k) \frac{1}{q_l - 1} \sum_{i=1}^{q_k - 1} \mathbb{E}\left(\sqrt{\frac{X_i^{(k)}}{X_0^{(k)}}} \right)
\end{aligned}$$

To shorten the notations we can omit the index of iteration t. Moreover, in the remainder of this proof, we choose to use simpler notations although not fully rigorous: $\mathbf{R}^{(j,l)}$ denotes a message going into a check node of degree j in $G(q_l)$ while $\mathbf{R}^{(i,k)}$ denotes a message going out of a variable of degree i in $G(q_k)$. However, there is not ambiguity in the sequel thanks to the unique use of indexes i, j, k, l and we always precise the nature of a message.

The n^{th} component of a message coming from a variable of degree i in $G(q_k)$ is denoted by $R_n^{(i,k)}$. The n^{th} component of the initial message going into a variable in $G(q_k)$ is denoted by $R_n^{(0)^{(k)}}$. The n^{th}

component of a message going into a degree i variable in $G(q_k)$ is denoted by $L_n^{(i,k)}$. The data pass, through a variable node of degree i in $G(q_k)$, is translated by

$$R_n^{(i,k)} = \mu R_n^{(0)^{(k)}} \prod_{p=1}^{i-1} L_n^{(i,k)} \ ,$$

where μ is a normalization factor, which has no impact in the sequel as only rates of vector components are involved. Let $\mathbf{R}_t^{(k)}$ denote the average message going out of a variable node in $G(q_k)$. By noting that the messages $\mathbf{L}^{(i,k)}$ are i.i.d. when (i,k) is set, we have:

$$\begin{aligned} D_a(\mathbf{R}_t^{(k)}) &= \sum_i \Pi(i|k) \frac{1}{q_k - 1} \sum_{n=1}^{q_k} \mathbb{E}\left(\sqrt{\frac{R_n^{(0)^{(k)}} \prod_{p=1}^{i-1} L_n^{(i,k)}}{R_0^{(0)^{(k)}} \prod_{p=1}^{i-1} L_0^{(i,k)}}} \right) \\ &= \sum_i \Pi(i|k) \frac{1}{q_k - 1} \sum_{n=1}^{q_k} \mathbb{E}\left(\sqrt{\frac{R_n^{(0)^{(k)}}}{R_0^{(0)^{(k)}}}} \right) \mathbb{E}\left(\sqrt{\frac{L_n^{(i,k)}}{L_0^{(i,k)}}} \right)^{i-1} \\ &= \sum_i \Pi(i|k) \frac{1}{q_k - 1} \sum_{n=1}^{q_k} \mathbb{E}\left(\sqrt{\frac{R_n^{(0)^{(k)}}}{R_0^{(0)^{(k)}}}} \right) D_a(\mathbf{L}^{(i,k)}) \end{aligned}$$

The last step is obtained thanks to the LM-invariance of $\mathbf{L}^{(i,k)}$. Finally we get:

$$D_a(\mathbf{R}_t^{(k)}) = D_a(\mathbf{R}^{(0)^{(k)}}) \sum_i \Pi(i|k) D_a(\mathbf{L}^{(i,k)}) \ . \tag{31}$$

Moreover, if we consider two LM-invariant vectors $\mathbf{L}^{(k)}$ and $\mathbf{L}^{(l)}$, where $\mathbf{L}^{(k)}$ is the random truncation of $\mathbf{L}^{(l)}$, it is clear that $D_a(\mathbf{L}^{(k)}) = D_a(\mathbf{L}^{(l)})$. Hence:

$$D_a(\mathbf{L}^{(i,k)}) = \sum_{j,l} \Pi(j,l|i,k) D_a(\mathbf{L}^{(j,l)}) \tag{32}$$

where $\mathbf{L}^{(j,l)}$ is the message going out of a check node of degree j in $G(q_l)$.

Let us recall the result of equation (68) in [19]:

$$1 - D(\underline{\mathbf{L}}_t) \geq \sum_d \rho_d \left(1 - D(\underline{\mathbf{R}}_t)\right)^{d-1} + O\left(D(\underline{\mathbf{R}}_t)^2\right) \ .$$

We can apply this result, since our definition of D_a corresponds to the definition the authors gave to D. We obtain

$$1 - D_a(\mathbf{L}^{(j,l)}) \geq (1 - D_a(\mathbf{R}^{(j,l)}))^{j-1} + O(D_a(\mathbf{R}^{(j,l)})^2) \tag{33}$$

where $\mathbf{R}^{(j,l)}$ is a message going into a check node of degree j in $G(q_l)$. It is straightforward from definition of $D_a(\cdot)$ to get:

$$D_a(\mathbf{R}^{(j,l)}) = \sum_{i',k'} \Pi(i',k'|j,l) \frac{q_{k'} - 1}{q_l - 1} D_a(\mathbf{R}^{(i',k')}) \ . \tag{34}$$

By gathering equations (31), (32), (33) and (34), we obtain:

$$D_a(\mathbf{R}_t^{(k)}) \leq D_a(\mathbf{R}^{(0)^{(k)}}) \sum_i \Pi(i|k) \left[\sum_{j,l} \Pi(j,l|i,k) \left(1 - \sum_{i',k'} \Pi(i',k'|j,l) \left(\frac{q_{k'}-1}{q_l-1} D_a(\mathbf{R}^{(i',k')}) \right)^{j-1} + O(D_a(\mathbf{R}^{(i',k')})^2) \right) \right]^{i-1} \quad (35)$$

which is also:

$$D_a(\mathbf{R}_t^{(k)}) \leq D_a(\mathbf{R}^{(0)^{(k)}}) \sum_i \Pi(i|k) \left[\sum_{j,l} \Pi(j,l|i,k) \left(1 - \sum_{i',k'} \Pi(i',k'|j,l) \frac{q_{k'}-1}{q_l-1} D_a(\mathbf{R}^{(i',k')}) \right)^{j-1} \right]^{i-1} + O(D_a(\mathbf{R}_{t-1})^2) \quad (36)$$

where $D_a(\mathbf{R}_{t-1}) = \sum_k D_a(\mathbf{R}_{t-1}^{(k)})$. By power series in the neighborhood of zero, we finally get:

$$D_a(\mathbf{R}_t^{(k)}) \leq D_a(\mathbf{R}^{(0)^{(k)}}) \Pi(i=2|k) \sum_{j,l} \Pi(j,l|i,k)(j-1) \sum_{k'} \Pi(k'|j,l) \frac{q_{k'}-1}{q_l-1} D_a(\mathbf{R}_{t-1}^{(k')}) + O(D_a(\mathbf{R}_{t-1})^2) \, . \quad (37)$$

Let $c^{(k)} = D_a(\mathbf{R}^{(0)^{(k)}})$ and $p(y|x)$ the transition probabilities of the memoryless output symmetric channel. We recall that we assume that the all-zero codeword has been sent. Then

$$\begin{aligned} c^{(k)} &= D_a(\mathbf{R}^{(0)^{(k)}}) \\ &= \frac{1}{q_k-1} \sum_{i=1}^{q_k-1} \mathbb{E} \left(\sqrt{\frac{p(y|i)}{p(y|0)}} \right) \\ &= \frac{1}{q_k-1} \sum_{i=1}^{q_k-1} \int \sqrt{\frac{p(y|i)}{p(y|0)}} p(y|0) dy \\ &= \frac{1}{q_k-1} \sum_{i=1}^{q_k-1} \int \sqrt{p(y|i)p(y|0)} dy \end{aligned}$$

We introduce hereafter some notations, for ease of reading:

Let $\mathbf{x}$ be a positive real-valued vector of size the number of different group orders. Let us define the g function by:

$$g(k, c^{(k)}, \Pi, \mathbf{x}) = c^{(k)} \Pi(i=2|k) \sum_{j,l} \Pi(j,l|i,k)(j-1) \sum_{k'} \Pi(k'|j,l) \frac{q_{k'}-1}{q_l-1} x_{k'} \, .$$

For more readable notations, we also define the vector output function $\mathbf{G}(\mathbf{x})$ by:

$$\mathbf{G}(\mathbf{x}) = \{ g(k, c^{(k)}, \Pi, \mathbf{x}) \}_k$$

which means that the p^{th} component of $\mathbf{G}(\mathbf{x})$ is $G_p(\mathbf{x}) = g(p, c^{(p)}, \Pi, \mathbf{x})$. Let us denote the convolution by $\otimes$. Then $\mathbf{x}^{\otimes n}$ corresponds to the convolution of vector $\mathbf{x}$ by itself n times. With these notations, we can write, for all $n > 0$:

$$D_a(\mathbf{R}_{t+n}^{(k)}) \leq g(k, c^{(k)}, \Pi, \mathbf{G}^{\otimes(n-1)}(\{ D_a(\mathbf{R}_t^{(k')}) \}_{k'})) + O(D_a(\mathbf{R}_t)^2) \, .$$

Let $P_e^{(k)^t} = P_e(\mathbf{R}_t^{(k)})$ be the probability that the message $\mathbf{R}_t^{(k)}$ be erroneous, i.e., corresponds to an incorrect decision. The average probability that any rightbound message be erroneous is $P_e^t = \sum_k \Pi(k) P_e^{(k)^t}$. Let us recall lemma (34) in [19]:

$$\frac{1}{q_k^2} D_a(\mathbf{X}^{(k)})^2 \leq P_e(\mathbf{X}^{(k)}) \leq (q_k - 1) D_a(\mathbf{X}^{(k)}) \ . \tag{38}$$

Let us consider a given k. If there exists a vector $\mathbf{x}$ such that $\lim_{n\to\infty} g(k, c^{(k)}, \Pi, \mathbf{G}^{\otimes(n-1)}(\mathbf{x})) = 0$, then there exist α and $n > 0$ such that if $\forall k, D_a(\mathbf{R}_{t_0}^{(k)}) < \alpha$, then

$$D_a(\mathbf{R}_{t_0+n}^{(k)}) < K_{k'} D_a(\mathbf{R}_{t_0}^{(k')}), \quad \forall k' \tag{39}$$

where, for all k', $K_{k'}$ is a positive constant smaller than 1. If we consider $P_e^{t_0} < \xi$ such that $\forall k, P_e^{(k)^{t_0}} < (q_k \alpha)^2$, then equation (38) ensures that $\forall k, D_a(\mathbf{R}_{t_0}^{(k)}) \leq \frac{\sqrt{P_e^{(k)^t}}}{q_k} < \alpha$.
As previously explained, in this case, there exits $n > 0$ such that inequation (39) is fulfilled. By induction, for all $t > t_0$, there exists $n > 0$ such that

$$D_a(\mathbf{R}_{t+n}^{(k)}) < K_{k'} D_a(\mathbf{R}_t^{(k')}), \quad \forall k' \ .$$

We have $\forall (k,t), D_a(\mathbf{R}_t^{(k)}) \geq 0$, therefore the sequence $\{D_a(\mathbf{R}_t^{(k)})\}_{t=t_0}^{\infty}$ converges to zero for all k. Finally, equation (38) ensures that, for all k, $P_e^{(k)^t}$ converges to zero as t tends to infinity. Thus, P_e^t, the global error probability, averaged over all symbol sizes, converges to zero as t tends to infinity.
This proves the sufficiency of the stability condition.

□

Lemma 6: *The connection between $x_{\mathbf{v}}$ and $x_{f(\mathbf{v})}$ is*

$$x_{ft(\mathbf{p})} = 1 - x_{\mathbf{p}} \ .$$

Proof: Let $\mathbf{p}$ be a probability vector of size q, associated to a symbol in $G(q)$, and $\mathbf{f}$ its Discrete Fourier Transform of size q too. p_k and f_i are the k^{th} and the i^{th} components of $\mathbf{p}$ and $\mathbf{f}$, respectively. $\mathbf{f}$ is defined by:

$$f_i = \sum_{k=0}^{q-1} p_k (-1)^{i \cdot k}, \qquad \forall i \in GF(q)$$

$i \cdot k$ is the scalar product between the binary representations of both elements i and k.
The mutual information I of a symmetric probability vector $\mathbf{p}$, under the all-zero codeword assumption,

is defined by

$$x_{\mathbf{p}} = 1 - \mathbb{E}_{\mathbf{p}} \left(\log_q(1 + \sum_{i=1}^{q-1} \frac{p_i}{p_0}) \right) .$$

As in the binary case, we want to prove that

$$x_{\mathbf{p}} = 1 - x_{\mathbf{f}}$$

where $x_{\mathbf{f}}$ is defined by $x_{\mathbf{f}} = 1 - \mathbb{E}_{\mathbf{f}} \left(\log_q(1 + \sum_{i=1}^{q-1} \frac{f_i}{f_0}) \right)$.

The equation $x_{\mathbf{p}} = 1 - x_{\mathbf{f}}$ is equivalent to

$$\begin{aligned}
\mathbb{E}_{\mathbf{f}} \left(\log_q(1 + \sum_{i=1}^{q-1} \frac{f_i}{f_0}) \right) &= 1 - \mathbb{E}_{\mathbf{p}} \left(\log_q(1 + \sum_{i=1}^{q-1} \frac{p_i}{p_0}) \right) \\
\mathbb{E}_{\mathbf{f}} \left(\log_q(1 + \sum_{i=1}^{q-1} \frac{f_i}{f_0}) \right) &= \mathbb{E}_{\mathbf{p}} \left(1 - \log_q(\frac{1}{p_0}) \right) \\
\mathbb{E}_{\mathbf{f}} \left(\log_q(1 + \sum_{i=1}^{q-1} \frac{f_i}{f_0}) \right) &= \mathbb{E}_{\mathbf{p}} \left(\log_q(qp_0) \right) \\
f_0 = 1 \quad &\text{implies} \\
\mathbb{E}_{\mathbf{f}} \left(\log_q(\sum_{i=0}^{q-1} f_i) \right) &= \mathbb{E}_{\mathbf{p}} \left(\log_q(qp_0) \right) \qquad (40)
\end{aligned}$$

Since $\sum_{i=0}^{q-1} f_i = \sum_{i=0}^{q-1} \sum_{k=0}^{q-1} p_j (-1)^{i \cdot k}$, it finally remains to prove that

$$\begin{aligned}
\sum_{i=0}^{q-1} \sum_{k=1}^{q-1} p_j (-1)^{i \cdot k} &= 0 \\
\sum_{k=1}^{q-1} p_j \sum_{i=0}^{q-1} (-1)^{i \cdot k} &= 0 \qquad (41)
\end{aligned}$$

which is ensured by

$$\sum_{i=0}^{q-1} (-1)^{i \cdot k} = 0, \qquad \forall k = \{1 \dots q-1\} .$$

We are going to demonstrate this last expression. Let say that k has m bits equal to 1 in its binary representation.

- m is even: $i \cdot k$ is

$$\text{even} \quad \frac{q}{2^m} \sum_{l=0}^{m/2} \binom{m}{2l} \quad \text{times} \qquad (42)$$

$$\text{odd} \quad \frac{q}{2^m} \sum_{l=0}^{m/2-1} \binom{m}{2l+1} \quad \text{times} \qquad (43)$$

- m is odd: $i \cdot k$ is

$$\text{even} \quad \tfrac{q}{2^m} \sum_{l=0}^{\frac{m-1}{2}} \binom{m}{2l} \quad \text{times} \tag{44}$$

$$\text{odd} \quad \tfrac{q}{2^m} \sum_{l=0}^{\frac{m-1}{2}} \binom{m}{2l+1} \quad \text{times} \tag{45}$$

We complete the proof by showing that equations (42) and (43) are equal, as well as equations (44) and (45):

$$(1-1)^m = \sum_{k=0}^{m} \binom{m}{k} (-1)^k = \sum_{l=0}^{a} \binom{m}{2l} - \sum_{l=0}^{b} \binom{m}{2l+1} = 0$$

where $a = \frac{m}{2}$ and $b = \frac{m}{2} - 1$ when m is even, and $a = b = \frac{m-1}{2}$ when m is odd. This completes the proof.

□

Detailed simulation results for rate one-half

SNR points: $[1.0, 1.2, 1.4, 1.6, 1.8, 1.9, 2.0]$;

Number of frames in error for hybrid LDPC code 1: $[100100100100623]$;

Number of sent frames for hybrid LDPC code 1: $[4101434928396721887900225315 0]$;

Number of frames in error for hybrid LDPC code 2: $[1001001005545829]$;

Number of sent frames for hybrid LDPC code 1: $[104366128379386450023180001231463001300523 50]$.

ACKNOWLEDGMENT

The authors would like to thank the reviewers and the associate editor for their many helpful suggestions. They would also like to thank Jean-Pierre Tillich and Charly Poulliat for their comments.
This work was funded by the French Armament Procurement Agency (DGA).

REFERENCES

[1] R. Gallager, "Low-density parity-check codes," PhD dissertation, MIT press, Cambridge, Massachusetts, 1963.

[2] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Transactions on Information Theory*, vol. 45, no. 2, pp. 399–431, 1999.

[3] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Transactions on Information Theory*, vol. 47, pp. 585–598, February 2001.

[4] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular LDPC codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 619–637, February 2001.

[5] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 599–618, February 2001.

[6] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: turbo-codes," *IEEE Transactions on Communications*, vol. 44, no. 10, pp. 1261–1271, October 1996.

[7] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

[8] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Transactions on Communications*, vol. 49, no. 10, pp. 1727–1737, October 2001.

[9] S. Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 db of the shannon limit," *IEEE Communications Letters*, vol. 5, pp. 58–60, February 2001.

[10] T. Etzion, A. Trachtenberg, and A. Vardy, "Which codes have cycle-free tanner graphs?" *IEEE Transactions on Information Theory*, vol. 45, no. 6, pp. 2173–2181, 1999.

[11] M. Chiani and A. Ventura, "Design and performance evaluation of some high-rate irregular low-density parity-check codes," in *Proceedings of IEEE Global Telecommunications Conference*, San Antonio, USA, November 2001.

[12] C. Di, R. Urbanke, and T. Richardson, "Weight distribution of low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 4839–4855, November 2006.

[13] J. Boutros, O. Pothier, and G. Zemor, "Generalized low density (Tanner) codes," in *Proceedings of IEEE Int. Conf. on Communications*, Vancouver, Canada, June 1999.

[14] E. Paolini, M. Fossorier, and M. Chiani, "Analysis of doubly-generalized LDPC codes with random component codes for the binary erasure channel," in *Proceedings of Allerton Conference on Communications, Control and Computing*, Monticello, USA, Sept 2006.

[15] I. Andriyanova, "Analysis and design of a certain family of graph-based codes: TLDPC," PhD dissertation, Ecole Nationale Supérieure des Télécommunications, Paris, France, 2006.

[16] M. Davey and D. MacKay, "Low density parity check codes over GF(q)," *IEEE Communications Letters*, vol. 2, no. 6, pp. 165–167, June 1998.

[17] M. Davey, "Error-correction using low density parity check codes," PhD dissertation, University of Cambridge, Cambridge, UK, December 1999.

[18] X.-Y. Hu and E. Eleftheriou, "Binary representation of cycle Tanner-graph GF(2^q) codes," in *Proceedings of IEEE International Conference on Communications*, Paris, France, June 2004, pp. 528–532.

[19] A. Bennatan and D. Burshtein, "Design and analysis of nonbinary LDPC codes for arbitrary discrete-memoryless channels," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 549–583, February 2006.

[20] A. Goupil, M. Colas, G. Gelle, and D. Declercq, "FFT-based BP decoding of general LDPC codes over Abelian groups," *IEEE Transactions on Communications*, vol. 55, no. 4, pp. 644–649, April 2007.

[21] D. Sridhara and T. Fuja, "Low density parity check codes over groups and rings," in *Proceedings of IEEE Information Theory Workshop*, Bangladore, India, October 2002.

[22] J. Boutros, A. Ghaith, and Y. Yuan-Wu, "Non-binary adaptive LDPC codes for frequency selective channels: code construction and iterative decoding," in *Proceedings of IEEE Information Theory Workshop*, Chengdu, China, October 2006.

[23] E. Paolini, "Iterative decoding methods based on low-density graphs," PhD dissertation, Universita degli studi di Bologna, Bologna, Italia, 2007.

[24] K. Kasai, T. Shibuya, and K.Sakaniwa, "Detailedly represented irregular LDPC codes," *IEICE Transactions on Fundamentals*, vol. E86-A, no. 10, pp. 2435–2443, October 2003.

[25] T. Richardson and R. Urbanke, "Multi-edge type LDPC codes," *available online*, April 2004.

[26] A. Voicila, D. Declercq, F. Verdier, M. Fossorier, and P. Urard, "Low complexity, low memory EMS algorithm for non-binary LDPC codes," in *Proceedings of IEEE International Conference on Communications*, Glasgow, UK, June 2007.

[27] J. Chen, A. Dholakia, E. Eleftheriou, M. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Transactions on Communications*, vol. 53, no. 8, pp. 1288–1299, August 2005.

[28] G. Li, I. Fair, and W. Krzymien, "Analysis of nonbinary LDPC codes using Gaussian approximation," in *Proceedings of IEEE International Symposium on Information Theory*, Yokohama, Japan, July 2003.

[29] S. Chung, T. Richardson, and R. Urbanke, "Analysis of sum-product decoding LDPC codes using a Gaussian approximation," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 657–670, February 2001.

[30] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Transactions on Communications*, vol. 52, pp. 670–678, April 2004.

[31] G. Liva, S. Song, L. Lan, Y. Zhang, S. Lin, and W. E. Ryan, "Design of LDPC codes: a survey and new results," *to appear in Journal on Communication Software and Systems*, 2006, available online.

[32] C. Poulliat, M. Fossorier, and D. Declercq, "Design of regular (2,dc)-LDPC codes over GF(q) using their binary images," *accepted in IEEE Transactions on Communications*, 2008.

[33] X.-Y. Hu, E. Eleftheriou, and D. Arnold, "Regular and irregular progressive edge-growth Tanner graphs," *IEEE Transactions on Information Theory*, vol. 51, pp. 386–398, January 2005.

[34] A. Brouwer and T. Verhoeff, "An updated table of minimum distance for binary linear codes," *IEEE Transactions on Information Theory*, vol. 39, no. 2, pp. 662–677, March 1993.

[35] D. Divsalar, C. Jones, S. Dolinar, and J. Thorpe, "Protograph based LDPC codes with minimum distance linearly growing with block size," in *Proceedings of IEEE Global Telecommunications Conference*, St. Louis, USA, November 2005.

[36] A. Venkiah, D. Declercq, and C. Poulliat, "Design of cages with a randomized progressive edge growth algorithm," *IEEE Communications Letters*, vol. 12, no. 4, pp. 301–303, February 2008.

[37] G. Yue, L. Ping, and X. Wang, "Generalized low-density parity-check codes based on Hadamard constraints," *IEEE Transactions on Information Theory*, vol. 53, no. 3, pp. 1058–1079, March 2007.

[38] L. Ping, W. Leung, and K. Wu, "Low-rate Turbo-Hadamard codes," *IEEE Transactions on Information Theory*, vol. 49, no. 12, pp. 3213–3224, December 2003.

[39] G. Yue, W. Leung, L. Ping, and X. Wang, "Low rate concatenated Zigzag-Hadamard codes," in *Proceedings of International Conference on Communications*, Istanbul, Turkey, June 2006.

[40] G. Bosco and S. Benedetto, "Soft decoding in optical systems: Turbo product codes vs. ldpc codes," in *Springer US - Optical Communication Theory and Techniques*, vol. Springer US, 2004, pp. 79–86.

[41] N. Shimanuki, B. Kurkoski, K. Yamagichi, and K. Kobayashi, "Improvements and extensions of low-rate Turbo-Hadamard codes," in *Proceedings of ISITA*, Seoul, Korea, October 2006.

[42] G. Liva, W. Ryan, and M. Chiani, "Quasi-cyclic generalized LDPC codes with low error floors," *IEEE Transaction on Communications*, vol. 56, no. 1, pp. 49–57, January 2008.

[43] X.-Y. Hu and M. Fossorier, "On the computation of the minimum distance of low-density patity-check codes," in *Proceedings of IEEE International Conference on Communications*, Paris, June 2004.

[44] T. Richardson, "in review of this paper," 2008.

[45] G. Yue, L. Ping, and X. Wang, "Low-rate generalized LDPC codes with Hadamard constraints," in *Proceedings of IEEE International Symposium on Information Theory*, Adelaide, Australia, September 2005.