A New Density Evolution Approximation for LDPC and Multi-Edge Type LDPC Codes

Sachini Jayasooriya, Student Member, IEEE, Mahyar Shirvanimoghaddam, Member, IEEE,
Lawrence Ong, Member, IEEE, Gottfried Lechner, Member, IEEE, and Sarah J. Johnson, Member, IEEE

Abstract—This paper considers density evolution for low-density parity-check (LDPC) and multi-edge type low-density parity-check (MET-LDPC) codes over the binary-input additive white Gaussian noise channel. We first analyze three single-parameter Gaussian approximations for density evolution and discuss their accuracy under several conditions, namely at low rates, with punctured and degree-one variable nodes. We observe that the assumption of symmetric Gaussian distribution for the density-evolution messages is not accurate in the early decoding iterations, particularly at low rates and with punctured variable nodes. Thus single-parameter Gaussian approximation methods produce very poor results in these cases. Based on these observations, we then introduce a new density evolution approximation algorithm for LDPC and MET-LDPC codes. Our method is a combination of full density evolution and a single-parameter Gaussian approximation, where we assume a symmetric Gaussian distribution only after density-evolution messages closely follow a symmetric Gaussian distribution. Our method significantly improves the accuracy of the code threshold estimation. Additionally, the proposed method significantly reduces the computational time of evaluating the code threshold compared to full density evolution thereby making it more suitable for code design.

Index Terms—Belief-propagation, density evolution, Gaussian approximation, low-density parity check (LDPC) codes, multi-edge type LPDC codes.

I. INTRODUCTION

Graph-based codes, such as low-density parity-check (LDPC), turbo, and repeat-accumulate codes, together with belief propagation (BP) decoding have shown to perform extremely close to the Shannon limit with reasonable decoding complexity [1]. These graph-based codes can be represented by a bipartite Tanner graph in which the variable and check nodes respectively correspond to the codewords symbols and the parity check constraints [2]. The error-correcting performance of a code is mainly characterized by the connectivity among the nodes in the Tanner graph where the node degree plays an important role. To specify the node degree distribution in the Tanner graph, the concept of degree distribution in either node perspective or edge perspective is introduced [3]. A code ensemble is then defined as the set of all codes with a particular degree distribution. As a unifying framework for graph-based codes, Richardson and Urbanke [4] proposed multi-edge type low density parity-check (MET-LDPC) codes. The benefit of the MET generalization is greater flexibility in the code structure, which can improve decoding performances. This generalization is particularly useful under traditionally difficult requirements, such as high-rate codes with very low error floors or low-rate codes [4].

A numerical technique, referred to as Density Evolution (DE), was formulated to analyze the convergence behavior of the BP decoder (i.e., the code threshold) for a given LDPC [5] or MET-LDPC [6] code ensemble under BP decoding, where the code threshold is defined as the maximum channel noise level at which the decoding error probability converges to zero as the code length goes to infinity. DE determines the performance of BP decoding for a given code ensemble by tracking the probability density function (PDF) of messages passed along the edges in the corresponding Tanner graph through the iterative decoding process. Then, it is possible to test whether, for a given channel condition and a given degree distribution, the decoder can successfully decode the transmitted message (with the decoding error probability tends to zero as the iterations progress). This allows us to design and optimize LDPC and MET-LDPC degree distributions using the DE threshold (i.e., the code threshold found using DE) as the cost function.

Calculating thresholds and designing LDPC and MET-LDPC degree distributions using DE are computationally intensive as they require numerical evaluations of the PDFs of the messages passed along the Tanner graph edges in each decoding iteration. Because of this, for LDPC codes on the binary input additive white Gaussian noise (BI-AWGN) channel, Chung et al. [6], [7] and Lehmann and Maggio [8] approximated the message PDFs by Gaussian PDFs, each using a single parameter, to simplify the analysis. Existing work concerning Gaussian approximations has relied on four different parameters in order to obtain a single-parameter model of the message PDFs, including mean value of the PDF [6], bit-error rate (BER) [8], reciprocal-channel approximation (RCA) [7] and mutual information (e.g., EXIT charts) [9]. Several papers [6], [10]–[12] have investigated the accuracy of the Gaussian approximation for BP decoding of standard LDPC codes and shown that it is accurate for medium-to-high rates. However in most of the literature regarding DE for MET-LDPC codes, only the full density evolution (full-DE) has been studied [5]. In full-DE, the quantized PDFs of the messages are passed along the edges without any approximation. Typically, for full-DE, thousands of points are used to accurately describe one message PDF. Schmalen and Brink [13] have used the Gaussian approximation based on the mean of the message PDF [6] to evaluate the behavior of protograph based LDPC codes, which is a subset of MET-LDPC codes.

The contributions of this paper are as follows: 1) We investigate the accuracy of Gaussian approximations for BP decoding. We follow the approximation techniques suggested
for LDPC codes [6–8], which describe each DE-message PDF using a single parameter. Based on our observations of the evolution of PDFs in the MET-LDPC codes, we found that those Gaussian approximations are not accurate for the scenarios where MET-LDPC codes are useful, i.e., at low rate and with punctured variable nodes. 2) In light of this, we propose a hybrid-DE method, which combines the full-DE and a Gaussian approximation. Our proposed hybrid-DE allows us to evaluate the code threshold (i.e., the cost function in the code optimization) of MET-LDPC and LDPC code ensembles significantly faster than the full-DE and with accuracy better than Gaussian approximations.

3) We design good MET-LDPC codes using the proposed hybrid-DE and show that the hybrid-DE well approximates the full-DE for code design.

This paper is organized as follows. Section II briefly reviews the basic concepts of MET-LDPC codes. In Section III, we extend Gaussian approximations for LDPC codes to MET-LDPC codes, and in Section IV, we discuss the accuracy of the Gaussian approximations under the conditions where MET-LDPC codes are more beneficial. Section V presents the proposed hybrid-DE method, and Section VI demonstrates the benefits of code design using the proposed hybrid-DE method over existing Gaussian approximations. Finally, Section VII concludes the paper.

II. BACKGROUND OF MET-LDPC CODES

A. MET-LDPC code ensemble

Unlike standard LDPC code ensembles where the graph connectivity is constrained only by the node degrees, in the multi-edge setting, several edge-types can be defined, and every node is characterized by the number of connections to edges of each edge-type. Within this framework, the degree distribution of MET-LDPC code ensemble can be specified through two node-perspective multinomials related to the variable nodes and check nodes respectively [5] page 383:

\[
L(r, x) = \sum_{b, d, r} L_{b,d} x^d r^b \quad (1)
\]

\[
R(x) = \sum_{R_d} R_d x^d, \quad (2)
\]

where \(b, d, r\) and \(x\) are vectors defined as follows. Let \(m_c\) denote the number of edge-types corresponding to the graph and \(m_v\) denote the number of different channels over which codeword bits may be transmitted. A vector \(d = [d_1, \ldots, d_{m_c}] \) is defined for each node in the graph, where \(d_i\) is the number of edges of the \(i\)th edge-type connected to that node, and we use \(x^d\) to denote \(\prod_{i=1}^{m_c} x_i^{d_i}\). As the variable nodes receive information from the channel over which the codeword bits are transmitted, there is an additional vector \(r^b = \prod_{i=1}^{m_v} r_i^{b_i}\) associated with each variable node where \(b_i\) designates the type of the message (i.e., the message PDF) it receives from the channel. Typically, \(b = [b_1, \ldots, b_{m_v}]\) has only two entries since in a BI-AWGN channel, a codeword bit is either punctured (the codeword bits not transmitted: \(b = [1, 0]\)) or transmitted through a single channel (\(b = [0, 1]\)). Finally, \(L_{b,d}\) and \(R_d\) are non-negative real numbers corresponding to the fraction of variable nodes of type \((b, d)\) and the fraction of check nodes of type \(d\) in the ensemble, respectively.

Node-perspective degree distributions can be converted to edge-perspective via the following multinomials, where \(\lambda_i(r, x)\) and \(\rho_i(x)\) are related to the variable nodes and check nodes, respectively [5] pages 390–391:

\[
\lambda_i(r, x) = \left( \frac{L_{x_1}(r, x)}{x_1(1, 1)} \frac{L_{x_2}(r, x)}{x_2(1, 1)} \cdots \frac{L_{x_{m_c}}(r, x)}{x_{m_c}(1, 1)} \right) \quad (3)
\]

\[
\rho_i(x) = \left( \frac{R_{x_1}(1)}{R_{x_1}(1)} \frac{R_{x_2}(1)}{R_{x_2}(1)} \cdots \frac{R_{x_{m_c}}(1)}{R_{x_{m_c}}(1)} \right), \quad (4)
\]

where

\[
L_{x_1}(r, x) = \frac{\partial}{\partial x_1} L(r, x)
\]

\[
L_{x_1}(1, 1) = \frac{\partial}{\partial x_1} L(r, x) \bigg|_{r=1, x=1}
\]

\[
R_{x_1}(x) = \frac{\partial}{\partial x_1} R(x)
\]

\[
R_{x_1}(1) = \frac{\partial}{\partial x_1} R(x) \bigg|_{x=1}
\]

The rate of a MET-LDPC code is given by

\[
r = L(1, 1) - R(1), \quad (5)
\]

where 1 denotes a vector of all 1’s with the length determined by the context.

A rate 1/2 MET-LDPC code ensemble is shown in Fig. 1 where the node-perspective degree distributions are given by

\[
L(r, x) = 0.5 r_1 x_1^2 + 0.3 r_1 x_1^3 + 0.2 r_2 x_2 x_3^2 + 0.2 r_3 x_3^2
\]

and

\[
R(x) = 0.4 r_1 x_1^2 + 0.1 r_3 x_3^2 + 0.2 r_4 x_4^2.
\]

Here \(r_0\) denotes punctured nodes and \(r_1\) denotes unpunctured nodes.

B. BP decoding and density evolution for MET-LDPC codes

In the BP decoding algorithm, messages are passed along the edges of the Tanner graph from variable nodes to their incident check nodes and vice versa until a valid codeword is found or a predefined maximum number of decoding iterations has been reached. Each BP decoding iteration involves two steps.

1) A variable node processes the messages it receives from its neighboring check nodes and from its corresponding
channel and outputs messages to its neighboring check nodes.
2) A check node processes inputs from its neighboring variable nodes and passes messages back to its neighboring variable nodes.

In most cases of binary codes transmitted on BI-AWGN channel, these BP decoding messages are expressed as log-likelihood ratios (LLRs) [14] pages 213-226. This is used to reduce the complexity of the BP decoder, as multiplication of message probabilities corresponds to the summation of corresponding LLRs.

Let $v_e^{(\ell)}$ denote the message LLR sent by a variable node to a check node along edge $e$, (i.e., variable-to-check) at the $\ell$th iteration of the BP decoding, and $u_v^{(\ell)}$ denote the message LLR sent by a check node to a variable node along edge $e$, (i.e., check-to-variable) at the $\ell$th iteration of the BP decoding. At the variable node, $v_e^{(\ell)}$ is computed based on the observed channel LLR ($u_0$) and message LLRs received from the neighboring check nodes except for the incoming message on the current edge for which the output message is being calculated. Thus the variable-to-check message on edge $e$ at the $\ell$th decoding iteration is as follows:

$$v_e^{(\ell)} = \begin{cases} u_0 & \text{if } \ell = 1, \\ u_0 + \sum_{i \neq e} u_i^{(\ell-1)} & \text{if } \ell > 1. \end{cases}$$

The message outputs on edge $e$ of a check node at the $\ell$th decoding iteration can be obtained from the “tanh rule”:

$$u_e^{(\ell)} = 2 \tanh^{-1} \left( \prod_{j \neq e} \tanh \left( \frac{v_j^{(\ell)}}{2} \right) \right).$$

For more details we refer readers to Ryan and Lin [14] pages 201-248.

DE is the main tool for analyzing the average asymptotic behavior of the BP decoder for MET-LDPC code ensembles, when the block length goes to infinity. For BP decoding on a BI-AWGN channel, these LLR values (i.e., $u_v, u_e, u_0$) are continuous random variables, thus can be described by PDFs for analysis using DE. To analyze the evolution of theses PDFs in the BP decoder, we define $f(v_k^{(\ell)}), f(u_k^{(\ell)}), f(u_0)$ which denote the PDF of the variable-to-check message, check-to-variable message and channel LLR, respectively. Unlike standard LDPC codes, in the MET framework, because of the existence of multiple edge-types, only the incoming messages from same edge-type are assumed to be identically distributed. However, all the incoming messages are assumed to be mutually independent. Recall that in MET-LDPC codes, a variable node is identified by its type, $(b,d)$, and a check node by its type, $d$. Thus from (6) the PDF of the variable-to-check message from a variable node type, $(b,d)$, along edge-type $i$ at the $\ell$th decoding iteration can be written as follows [5] pages 390-391:

$$f(v_{b,d}^{(\ell)}(i)) = f(u_b) \bigotimes_{k=1,k \neq i} ^{m_d} f(u_{\ell-1}^{(\ell)}(k))) \bigotimes_{d} d_k, \quad (8)$$

where $\bigotimes$ denotes convolution. The $d_1$-fold and $(d_i - 1)$-fold convolutions follow from the assumption that the incoming messages from a check node along edge-type $i$ are independent and identically distributed and $f(u_{\ell-1}^{(\ell)}(i))$ is used to denote this common PDF. $f(u_b)$ is the PDF of the channel LLR.

From (7) the PDF of the check-to-variable message from a check node type, $d$, along edge-type $i$ at the $\ell$th decoding iteration can be calculated as follows [5] pages 390-391:

$$f(v_d^{(\ell)}(i)) = \left[ f(v_{\ell}^{(\ell)}(i))) \right] \bigotimes_{k=1,k \neq i} (d_i - 1) m_d f(v_{\ell}^{(\ell)}(k))) \bigotimes_{d} d_k, \quad (9)$$

where $f(v_{\ell}^{(\ell)}(i))$ is the PDF of the message from a variable node along edge-type $i$ at the $\ell$th decoding iteration. The computation of $f(v_{\ell}^{(\ell)}(i))$ is not as straightforward as that for $f(v_{\ell}^{(\ell)}(i))$ and requires the transformation of $\log(\cdot)$ and $\log^{-1}(\cdot)$. So we use $\bigotimes$ to denote the convolution when computing the PDF of $f(v_{\ell}^{(\ell)}(i))$ for check-to-variable messages. For more details we refer readers to Richardson and Urbanke [5] pages 390-391, 459-478.

### III. GAUSSIAN APPROXIMATIONS TO DENSITY EVOLUTION FOR MET-LDPC CODES

In this section, we consider MET-LDPC codes over BI-AWGN channels with Gaussian approximations to DE. As already shown for LDPC codes [6], [11], the PDFs of variable-to-check and check-to-variable messages can be close to a Gaussian distribution in certain cases, such as when check node degrees are small and variable node degrees are relatively large. Since a Gaussian PDF can be completely specified by its mean ($m$) and variance ($\sigma^2$), we need to track only these two parameters during the BP decoding algorithm. Furthermore, it was shown by Richardson et al. [1] that the PDFs of variable-to-check and check-to-variable messages and channel inputs satisfy the symmetry condition: $f(x) = e^{\sigma^2} f(-x)$ where $f(x)$ is the PDF of variable $x$. This condition greatly simplifies the analysis because it implies $\sigma^2 = 2m$ and reduces the description of the PDF to a single parameter. Thus, by tracking the changes of the mean ($m$) during iterations, we can determine the evolution of the PDF of the check node message, $f(u_{\ell}^{(\ell)}(i)) = N(m_{\ell}^{(\ell)}, 2m_{\ell}^{(\ell)})$ and the variable node message, $f(v_{\ell}^{(\ell)}) = N(m_{\ell}^{(\ell)}, 2m_{\ell}^{(\ell)})$ where $N(m, \sigma^2)$ is the Gaussian PDF with mean $m$ and variance $\sigma^2$. $m_{\ell}^{(\ell)}$ and $m_{\ell}^{(\ell)}$ are the mean of the variable-to-check and check-to-variable messages, respectively.

#### A. Approximation 1: Gaussian approximation based on the mean of the message PDF

In this subsection, we will extend the Gaussian approximation method proposed by Chung et al. [6] for the threshold estimation of standard LDPC codes to that of MET-LDPC codes. This method is based on approximating message PDFs using a single parameter, i.e., the mean of the message PDF.

---

1Throughout this paper, we assume that the all-zero codeword is sent.
2The BP decoding messages received by every node at every iteration are independent and identically distributed.
Recall that a variable node is identified by its type, \((b, d)\), and a check node by its type, \(d\). Since the PDFs of the messages sent by the variable node are approximated as Gaussian, the mean of the variable-to-check message from a variable node type, \((b, d)\), along edge-type \(i\) at the \(\ell\)th decoding iteration is given by

\[
m_{b,d}^{(\ell)}(i) = m_{ub} + (d_i - 1)m_{ui}^{(\ell-1)}(i) + \sum_{k=1, k \neq i}^{m_e} d_k m_{ui}^{(\ell-1)}(k),
\]

where \(m_{ub}\) is the mean of the message from the channel and \(m_{ui}^{(\ell-1)}(i)\) is the mean of the check-to-variable message along edge-type \(i\) at the \((\ell-1)\)th decoding iteration. The updated mean of the check-to-variable message from check node type \(d\) along edge-type \(i\) at the \(\ell\)th decoding iteration can be written as

\[
m_{ud}^{(\ell)}(i) = \phi^{-1}(1 - \left[1 - \phi(m_{ui}^{(\ell)}(i))\right]^{d_i - 1} \prod_{k=1, k \neq i}^{m_e} \left[1 - \phi(m_{ui}^{(\ell)}(k))\right]^{d_k}),
\]

where \(m_{ui}^{(\ell)}(i)\) is the mean of the variable-to-check message along edge-type \(i\) at the \(\ell\)th decoding iteration. The mean of the variable-to-check and the check-to-variable messages along edge-type \(i\) at the \(\ell\)th decoding iteration is given by

\[
m_v^{(\ell)}(i) = \sum_d \lambda_i d m_{vd}^{(\ell)}(i)
\]

\[
m_u^{(\ell)}(i) = \sum_d \rho_i d m_{ud}^{(\ell)}(i),
\]

where \(\lambda_i\) and \(\rho_i\) are the variable and check node edge-degree distributions with respect to edge-type \(i\), respectively and

\[
\phi(x) = \begin{cases} 
1 - \frac{1}{\sqrt{4\pi x}} \int_R \tanh(\frac{\sqrt{x}}{2}) e^{-(u-x)^2/(4x)} du, & \text{if } x > 0 \\
1, & \text{otherwise.} 
\end{cases}
\]

It is important to note that \(\phi(x)\) is continuous and monotonically decreasing over \([0, \infty)\) with \(\phi(0) = 1\) and \(\phi(\infty) = 0\) [6].

### B. Approximation 2: Gaussian approximation based on the bit error rate

In this subsection, we will extend a Gaussian approximation method proposed by Lehmann et al. [8] that estimates thresholds of standard LDPC codes to that of MET-LDPC codes. This method is based on a closed-form expression in terms of error probabilities (i.e., the probability that a variable node is sending an incorrect message).

Consider a check node of type, \(d\). The error probability of a check-to-variable message from a check node type, \(d\) along edge-type \(i\) at the \(\ell\)th decoding iteration is given by

\[
P_{vd}^{(\ell)}(i) = \frac{1}{2} \left[1 - (1 - 2P_v^{(\ell-1)}(i))^{d_i - 1} \prod_{k=1, k \neq i}^{m_e} (1 - 2P_v^{(\ell-1)}(k))^{d_k}\right],
\]

where \(P_v^{(\ell-1)}(i)\) is the average error probability of the variable-to-check message along edge-type \(i\) at the \((\ell-1)\)th decoding iteration. Since we suppose that the all-zero codeword is sent, the error probability of a variable node at the \(\ell\)th decoding iteration is simply the average probability that the variable-to-check messages are negative. We also assume that the PDF of variable-to-check message is symmetric Gaussian; therefore the error probability of a variable-to-check message from a variable node type, \((b, d)\) along edge-type \(i\) at the \(\ell\)th decoding iteration is given by

\[
P_{vb,d}^{(\ell)}(i) = Q\left(\sqrt{\frac{m_{vb,d}^{(\ell)}(i)}{2}}\right),
\]

where \(m_{vb,d}^{(\ell)}\) is the mean of the variable-to-check message from a variable node type, \((b, d)\) along edge-type \(i\) at the \(\ell\)th decoding iteration, and

\[
Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-t^2/2} dt.
\]

\(m_{vb,d}^{(\ell)}\) can be calculated using (10) by substituting

\[
m_{u}^{(\ell)}(i) = 2 \left(Q^{-1}(P_{u}^{(\ell)}(i))\right)^2
\]

for each \(m_{u}^{(\ell)}(i)\), where \(P_{u}^{(\ell)}(i)\) is the average error probability of the check-to-variable message along edge-type \(i\) at the \(\ell\)th decoding iteration. The average error probability of the variable-to-check and the check-to-variable messages along edge-type \(i\) at the \(\ell\)th decoding iteration is given by

\[
P_{v}^{(\ell)}(i) = \sum_d \lambda_i d P_{vd}^{(\ell)}(i)
\]

\[
P_{u}^{(\ell)}(i) = \sum_d \rho_i d P_{ud}^{(\ell)}(i),
\]

where \(\lambda_i\) and \(\rho_i\) are the variable and check node edge-degree distributions with respect to edge-type \(i\), respectively.

### C. Approximation 3: Gaussian approximation based on the reciprocal-channel approximation

In this subsection, we will extend another Gaussian approximation method, proposed by Chung [7] pages 189-193] to estimates thresholds of regular LDPC codes, to that of MET-LDPC codes. This method is called reciprocal-channel approximation (RCA), which is based on reciprocal-channel mapping and mean (\(m\)) of the node message is used as the one-dimensional tracking parameter for the BI-AWGN channel.

With the RCA technique in DE, \(m\) is additive at the variable nodes similar to Approximation 1 (see [10]). The difference between Approximation 1 and Approximation 3 is how the check nodes calculate their output messages. Instead of evaluating the \(\tanh\) function in Approximation 1, Approximation 3 uses the reciprocal-channel mapping, \(\psi(m)\), which is additive at the check nodes. \(\psi(m)\) is defined as follows [7] pages 189-193]:

\[
\psi(m) = C_{AWGN}^{-1}(1 - C_{AWGN}(m)),
\]

where \(C_{AWGN}(m)\) is the capacity of the BI-AWGN channel as a function of the mean of the channel message, and

\[
C_{AWGN}(m) = 1 - \frac{1}{2\sqrt{\pi m}} \int_{-\infty}^{\infty} \log_2(1 + e^{-x}) e^{-\frac{(x-m)^2}{m}} dx,
\]
Then the mean of check-to-variable message from a check node type, \( \ell \) along edge-type \( i \) at the \( \ell \)th decoding iteration is given by
\[
\psi(m_{u}(i)(\ell)) = (d_{i} - 1)\psi(m_{v}(i)(\ell)) + \sum_{k=1,k \neq i}^{m_{\ell}} d_{k}\psi(m_{u}(k)).
\]
(21)

\( m_{v}(i)(\ell) \) and \( m_{u}(i)(\ell) \) can be calculated from (12) and (13), respectively.

IV. VALIDITY OF THE GAUSSIAN ASSUMPTION FOR DENSITY EVOLUTION

As we discussed in Section II, in the BP decoder, there are three types of messages: the channel message, the variable-to-check message, and the check-to-variable message. We analyze the PDF of these messages on the BI-AWGN channel to evaluate the Gaussian assumption for DE message PDFs.

A. Channel messages

Let \( c = (c_{1}, c_{2}, \ldots) \) be a binary codeword \((c_{i} \in \{0, 1\})\) on a BI-AWGN channel. A codeword bit, \( c_{i} \) can be mapped to the transmitted symbol \( x_{i} = 1 \) if \( c_{i} = 0 \) and \( x_{i} = -1 \), otherwise. Then, the \( \ell \)th received symbol at the output of the AWGN channel is \( y_{i} = \sqrt{E_{c}}x_{i} + z_{i} \) where \( E_{c} \) is the energy per transmitted symbol and \( z_{i} \) is the AWGN, \( z_{i} \sim \mathcal{N}(0, \sigma_{n}^{2}) \). The LLR \((\ell)\) for the received signal, \( y_{i} \) is given by
\[
u_{0} = L(x_{i}|y_{i}) = \log \frac{Pr(y_{i}|x_{i} = 1)}{Pr(y_{i}|x_{i} = -1)} = 2\sqrt{E_{c}}/\sigma_{n}^{2}y_{i}.
\]
Assuming that the all-zero codeword is sent and that \( \sqrt{E_{c}} = 1 \),
\[
u_{0} = L(x_{i}|y_{i}) = \frac{2}{\sigma_{n}^{2}}y_{i},
\]
(22)
which is a Gaussian random variable with \( E[\nu_{0}] = \frac{2}{\sigma_{n}^{2}} \) and \( \text{Var}[\nu_{0}] = \frac{4}{\sigma_{n}^{4}} \). Since the variance is twice the mean, the channel message has a symmetric Gaussian distribution.

B. Variable-to-check messages

Consider the variable node update in (6). In the first iteration of the BP decoding, each variable node receives only a non-zero message from the channel. Hence the first set of messages passed from the variable nodes to the check nodes follow a symmetric Gaussian PDF. The following theorem describes the variable-to-check message exchanges in the \( \ell \)th, \( \ell > 1 \), iteration of the BP decoder.

Theorem 1: The PDF of the variable-to-check message at the \( \ell \)th decoding iteration \((v^{(\ell)})\) is a Gaussian distribution if all check-to-variable messages \((u^{(\ell)})\) are Gaussian. If \( u^{(\ell)} \) are not Gaussian then the PDF of \( v^{(\ell)} \) converges to a Gaussian distribution as variable node degree tends to infinity.

Proof: The update rule at a variable node in (6) is the summation of the channel message and incoming messages from check nodes \((u^{(\ell)})\). Since the channel is BI-AWGN, \( u_{0} \) follows a symmetric Gaussian distribution. If all \( u^{(\ell)} \)s (which are mutually independent) are Gaussian, then \( v^{(\ell)} \) is also Gaussian, because it is the sum of independent Gaussian random variables.

C. Check-to-variable messages

Before analyzing the check-to-variable messages, let us first state a few useful lemmas and definitions, upon which our analysis is based.

Definition 1: If the random variable \( X \) is Gaussian distributed and \( X = \ln(Y) \), then random variable \( Y \) is said to be lognormally distributed.

Lemma 1 (15): If \( x_{1}, x_{2}, \ldots, x_{n} \) are independent Gaussian random variables with means \( m_{1}, m_{2}, \ldots, m_{n} \) and variances \( \sigma_{1}^{2}, \sigma_{2}^{2}, \ldots, \sigma_{n}^{2} \), and \( \{a_{i}\} \) is a set of arbitrary non-zero constants, then the linear combination, \( Z = \sum_{i=1}^{n} a_{i} x_{i} \) follows a Gaussian distribution with mean \( \sum_{i=1}^{n} a_{i} m_{i} \) and variance \( \sum_{i=1}^{n} a_{i}^{2} \sigma_{i}^{2} \).

Lemma 2: Let \( Y \) be a lognormal random variable. Then \( (Y)^{a} \) follows a lognormal distribution, where \( a \in \mathbb{R} \).

Proof: Since \( Y \) is a lognormal random variable then from Definition 1 \( Y = e^{X} \) where \( X \) is Gaussian random variable. According to Lemma 1 \( aX \) also follows a Gaussian distribution. Thus from Definition 1 \( e^{aX} = (Y)^{a} \) follows a lognormal distribution.

Remark 2: The assumption for the Gaussian approximation is that the sum of \( N \) independent lognormal random variables can be well approximated by another lognormal random variable. This has been shown to be true for \( N = 2 \) (17).

Now consider the check node update in (7) at the \( \ell \)th decoding iteration.

Remark 3: The PDF of the check-to-variable message at the \( \ell \)th decoding iteration \((u^{(\ell)})\) is a Gaussian distribution provided that the variable-to-check messages are approximately Gaussian and reasonably reliable, and the degrees of the check nodes are small.

Consider the check node with degree \( d_{c} \). We can rewrite (7) as follows:
\[
u_{c}(\ell) = \prod_{j=1}^{d_{c}-1} \text{sgn}(u^{(j)}(\ell)) \varphi \left( \sum_{j=1}^{d_{c}-1} \varphi^{-1}(v^{(j)}(\ell)) \right)
\]
(23)
where we define \( \varphi(x) = \log(\tanh(\frac{x}{2})) = \log\left(\frac{e^{x} - 1}{e^{x} + 1}\right) \) for \( x \geq 0 \) and note that \( \varphi^{-1}(x) = \log\left(\frac{e^{x} + 1}{e^{x} - 1}\right) = \varphi(x) \).

3. Since the all zero codeword is transmitted, reasonably reliable messages suggest that majority of \( v^{(\ell)} \)’s take large positive values (i.e., \( v^{(\ell)} \) has a large mean).

4. Following Remark 3 by “small check node degree” we mean the check node degree equals two.
Suppose \( v_j^{(l)} \)'s are reasonably reliable. Using Taylor series expansion, \( \varphi(v) \) can be expressed as
\[
\varphi(v) = \log\left(\frac{e^v - 1}{e^v + 1}\right) = 2(e^v)^{-1} - \frac{2}{3}(e^v)^{-3} + \frac{2}{5}(e^v)^{-5} + \ldots
\]
For simplicity, we omitted the indices \( j, c \). Since \( v \) follows an approximate Gaussian distribution, according to Definition 1, \( e^v \) follows an approximate lognormal distribution and from Lemma 2 \((e^v)_{b_i}, b_i \in \mathbb{Z}\) follows an approximate lognormal distribution. Using the assumption that, the sum of a set of independent lognormal random variables is approximately lognormal when the set size is small (see Remark 2), \( \varphi(v) \) (in step 1) follows an approximate lognormal distribution. This is because, when \( v \) is large with high probability, the higher-order terms in the Taylor series expansion of \( \varphi(v) \) are insignificant compared to the first few terms. Next, the \( \varphi(v_j^{(l)}) \)'s are mutually independent. Thus, according to Remark 2, \( \sum_{j=1}^{d_c - 1} \varphi(v_j^{(l)}) \) (in step 2) follows an approximate lognormal distribution when the check node degree is small. Finally, from Definition 1 \( \varphi(\sum_{j=1}^{d_c - 1} v_j^{(l)}) \) (in step 3) will follow an approximate Gaussian distribution if the result of step 2 is lognormally distributed.

Remark 4: The assumption that, the sum of \( N \) independent lognormal random variables can be well approximated by another lognormal random variable, is not true when \( N \) is large.

Using the above results, we investigate the accuracy of Gaussian approximations to full-DE of low rate MET-LDPC codes, with punctured and degree-one variable nodes. These are the cases where MET-LDPC codes are most beneficial. We also evaluate full-DE simulations for these codes to measure how close the actual message PDF is (under the full-DE) to a Gaussian PDF using the Kullback-Leibler (KL) divergence \([18]\) as our measure. A small value of KL divergence indicates that actual PDF is close to a Gaussian PDF. We calculate the KL divergence between 1) the actual message PDF (under the full-DE) and a Gaussian PDF with the same mean and variance to check whether it follows a Gaussian distribution, 2) the actual message PDF (under the full-DE) and a symmetric Gaussian PDF with the same mean to check whether it follows a symmetric Gaussian distribution.

1) Low SNR: In the case of standard LDPC codes, it has been observed \([10]-[12]\) the check-to-variable messages significantly deviate from a symmetric Gaussian distribution at low signal-to-noise ratios (SNR), even if the variable-to-check messages are close to a Gaussian distribution. Thus Gaussian approximations based on single-parameter models do not perform well for the codes at low SNRs. Here we explain the reason behind this, based on the assumptions required for Gaussian approximations to be accurate.

According to Remark 3, the PDF of the check-to-variable messages \( u^{(l)} \) can be well approximated by a Gaussian PDF, if the variable-to-check messages \( v_j^{(l)} \)'s are reasonably reliable given that \( v_j^{(l)} \)'s are approximately Gaussian and check node degrees are small. At low SNR, the initial \( v_j^{(l)} \)'s are not reasonably reliable. Thus \( u^{(l)} \) may not follow a Gaussian distribution in early decoding iterations. However, if the SNR is above the code threshold, the decoder converges to zero error probability as decoding iterations proceed, thus the PDF of \( v_j^{(l)} \) moves to right and the \( v_j^{(l)} \)'s become more reliable. Hence \( u^{(l)} \) may follow a Gaussian distribution at later decoding iterations.

To illustrate this via an example, we plot the actual message PDFs in the BP decoding and the KL divergence between the PDF of check-to-variable message from edge-type two and the corresponding symmetric Gaussian PDF (dotted line) of rate 1/10 MET-LDPC code with \( L(r, x) = 0.1r_1x_1r_2x_2^30.025r_1x_1^3x_2^5+0.875r_1x_3, R(x) = 0.025x_1^3+0.875x_2^3x_3 \) is compared with the symmetric Gaussian PDF of the same mean.

Fig. 3. KL divergence of check-to-variable message PDF from edge-type two to the corresponding Gaussian PDF (solid line) and to the corresponding symmetric Gaussian PDF (dotted line) of rate 1/10 MET-LDPC code with \( L(r, x) = 0.1r_1x_1r_2x_2^30.025r_1x_1^3x_2^5+0.875r_1x_3, R(x) = 0.025x_1^3+0.875x_2^3x_3 \) above the code threshold, the decoder converges to zero error probability as decoding iterations proceed, thus the PDF of \( v_j^{(l)} \) moves to right and the \( v_j^{(l)} \)'s become more reliable. Hence \( u^{(l)} \) may follow a Gaussian distribution at later decoding iterations.

2) Large check node degree: It has been observed \([10]-[12]\) that the check-to-variable messages significantly deviate from a symmetric Gaussian distribution when the check node degree is large, even if the variable-to-check messages are close to a Gaussian distribution. Thus single-parameter Gaussian approximation models do not perform well for the standard LDPC codes with large check node degrees. Here we explain
the reason behind this, based on the assumptions required for Gaussian approximations to be accurate.

According to Remark 3, the PDF of the check-to-variable messages \(v_j^{(t)}\)s can be well approximated by a Gaussian PDF, when check node degrees are small given that \(v_j^{(t)}\)s are approximately Gaussian and reasonably reliable. The assumption in step 2 (see Remark 4), that is the sum of \(N\) independent lognormal random variables can be well approximated by another lognormal random variable, is clearly not true if the check node degree is large (see Remark 4). Thus \(u^{(t)}\) may not follow a Gaussian distribution for larger the check node degrees.

To evaluate the combined effect of the SNR and the check node degree, we plot the KL divergence of check-to-variable message PDFs to the corresponding symmetric Gaussian PDFs for a rate 1/10 MET-LDPC code at the first decoding iteration for different SNRs and different check node degrees in Fig. 2. Our simulations show that with a large check node degree of 15, the KL divergence is large. Based on this we claim that single-parameter Gaussian approximations may not be a good approximation to DE for codes with large check node degrees.

3) The effect of punctured variable nodes: One of the modifications of MET-LDPC codes over standard LDPC codes is the addition of punctured variable nodes to improve the code threshold (a different use of puncturing than its typical use to increase the rate). We observe that these punctured nodes have a significant impact on the accuracy of the Gaussian approximation of both variable-to-check messages and check-to-variable messages. According to Theorem 1 if \(u_j^{(t)}\)'s are not Gaussian, the PDF of the variable-to-check messages \(v_j^{(t)}\) converges to a Gaussian distribution as the variable node degree tends to infinity. In the case of punctured nodes, \(u_0\) equals zeros as punctured bits are not transmitted through a channel. Hence in punctured variable nodes, \(v_j^{(t)}\) is equivalent to the sum of \(v_j^{(t)}\) only, which are heavily non Gaussian at early decoding iterations. Thus if the variable node degree is not large enough, then \(v_j^{(t)}\) from punctured variable nodes may not follow a Gaussian distribution at early decoding iterations.

The punctured variable nodes adversely affect the check-to-variable messages as well. According to Remark 3, the PDF of the check-to-variable messages \(u^{(t)}\) can be well approximated by a Gaussian PDF, if the variable-to-check messages \(v_j^{(t)}\) are following a Gaussian distribution at later decoding iterations. However in general, to become Gaussian it takes more decoding iterations than typical for a code without punctured nodes.

To illustrate the effect of punctured nodes via an example, we plot the KL divergence of variable-to-check and check-to-variable messages to the corresponding symmetric Gaussian PDFs of rate a 1/2 MET-LDPC code with punctured nodes in Fig. 3. It is clear from Fig. 3(a) that the KL divergence of the variable-to-check message to the corresponding symmetric Gaussian PDF from punctured nodes has a larger value than that from the unpunctured nodes in the same code. Fig. 3(b) shows the corresponding effect on the KL divergence of the check-to-variable messages. Furthermore, the decrease of the KL divergence with decoding iterations in Fig. 3 implies that \(v_j^{(t)}\) and \(u^{(t)}\) are following a Gaussian distribution at later decoding iterations. However in general, to become Gaussian it takes more decoding iterations than typical for a code without punctured nodes.

4) The effect of degree-one variable nodes: One of the advantages of the MET-LDPC codes is the addition of degree-one variable nodes to improve the code threshold. However we observe that degree-one variable nodes can affect the Gaussian approximation for the check-to-variable messages. Here we explain the reason behind this, based on the assumptions required for Gaussian approximations to be accurate.

According to Remark 3, the PDF of the check-to-variable
messages \( (w^{(ℓ)}) \) can be well approximated by a Gaussian PDF, if the variable-to-check messages \( (v^{(ℓ)}_j) \) are reasonably reliable given that \( v^{(ℓ)}_j \) are Gaussian. Even though \( v^{(ℓ)}_j \) received from edges connected to degree-one variable nodes are Gaussian, they are not reasonably reliable. This is because variable nodes of degree-one never update their \( v^{(ℓ)}_j \) as they do not receive information from more than one neighboring check node. So the \( w^{(ℓ)} \) may not follow Gaussian distribution. Consequently, this may reduce the validity of the Gaussian approximation to DE for MET-LDPC codes. Similarly any check node that receives input messages from a degree-one variable node never outputs a distribution with an infinitely large mean in any of its edge messages updated with information from degree-one variable nodes. This is because, if \( x_1 \ldots x_n \) are a set of independent random variables, then \( \mathcal{L}(x_1) \oplus \mathcal{L}(x_2) \oplus \cdots \oplus \mathcal{L}(x_n) = \mathcal{L}(x_1 \oplus x_2 \oplus \cdots \oplus x_n) \), and \( \mathcal{L}(x_1 \oplus x_2 \oplus \cdots \oplus x_n) \xrightarrow{P} \mathcal{L}(x_1) \) as \( \min_{2 \leq i \leq n} \mathbb{E}[\mathcal{L}(x_i)] \rightarrow \infty \) [19] pages 735-738, as is the case with degree-one variable nodes. We observed through the simulations that the degree-one variable nodes have a small impact on the accuracy of the Gaussian approximation of both variable-to-check messages and check-to-variable messages.

V. HYBRID DENSITY EVOLUTION FOR MET-LDPC CODES

All the Gaussian approximations we discussed in Section [3] are based on the assumption that the PDFs of the variable-to-check and the check-to-variable messages can be well approximated by symmetric Gaussian distributions. This assumption is quite accurate at the later decoding iterations, but less accurate in the early decoding iterations particularly at low SNRs or with punctured variable nodes or with large check node degrees as we have observed in Section [4]. Making the assumption of symmetric Gaussian distributions at the beginning of the DE calculation produces large errors between the estimated and true distributions. Even when the true distributions do become Gaussian, the approximations give incorrect Gaussian distributions due to the earlier errors. These errors propagate throughout the DE calculation and cause significant errors in the final code threshold result for MET-LDPC codes as we will see in Section [VI]. Through simulations, we observed that, when the channel SNR is above the code threshold, the PDFs of the node messages (i.e., variable-to-check and the check-to-variable messages) eventually do become symmetric Gaussian distributions as decoding iterations proceeds. This implies that making the assumption of symmetric Gaussian distributions in the later decoding iterations of the DE calculation is reasonable. This motivates us to propose a hybrid density evolution (hybrid-DE) algorithm for MET-LDPC codes which is a combination of the full-DE and the mean-based Gaussian approximation (Approximation 1). The key idea in hybrid-DE is that we do not assume that the node messages are symmetric Gaussian at the beginning of the DE calculation, i.e., hybrid-DE method initializes the node message PDFs using the full-DE and then switches to the Gaussian approximation.

There are two options for switching from the full-DE to the Gaussian approximation. As option one, we can impose a limitation for the number of maximum full-DE iterations in hybrid-DE, in which we do few full-DE iterations and then switch to the Gaussian approximation. Although this is the simplest option, it gives a nice trade-off between accuracy and efficiency of threshold computation as shown in Fig. [6]. The second option is that we can switch from full-DE to the Gaussian approximation after that the PDFs for the node messages become nearly symmetric Gaussian. The KL divergence [18] can be used to check whether a message PDF is close to a symmetric Gaussian distribution. Thus, as the second option, we can impose a limitation for the KL divergence between the actual node message PDF and a symmetric Gaussian PDF. This is a more accurate way of switching than option one. Because the value of the KL divergence depends on the shape of the PDF of the node messages, thus the switching point is changing appropriately with the condition (such as SNR, code rate) we are looking at.

Each option has its own pros and cons. For instance, if the channel SNR is well above the code threshold, the node messages are close to a symmetric Gaussian distribution before the imposed limit in option one for the full-DE iterations. Thus we are doing extra full-DE iterations that are not necessary. This reduces the benefit of hybrid-DE by adding extra runtime. In such a situation, we can introduce the second option (i.e., KL divergence limit) in addition to reduce run-time by halting the full-DE iterations once the PDF is sufficiently Gaussian. On the other hand, if the channel SNR is below the code threshold, the decoder never converge to a zero-error probability as decoding iterations proceed, thus node messages may not ever follow a symmetric Gaussian distribution. This makes the option two hybrid-DE always remain at full-DE, as it never meets the target KL divergence limit. Thus forcing a limitation for the number of full-DE iterations (i.e., option one) is required in order to improve the run-time of hybrid-DE. Because of these reasons, we can introduce both options to the hybrid-DE where option one acts as a hard limit and option two acts as a soft limit. This is a particularly beneficial way to do the trade-off between accuracy and efficiency when computing the code threshold. We found that it is possible to impose both options in the hybrid-DE to significantly improve computational time without significantly reducing the accuracy of the threshold calculation.

Throughout this paper, the check-to-variable message with the largest check node degree is chosen to check the KL divergence because the most significant errors relating to the estimation of the PDF occurs at large degree check nodes as we observed in Section [IV]. While running, the DE algorithm periodically calculates the KL divergence between the actual message PDF (under the full-DE) and a symmetric Gaussian PDF with the same mean for the selected check node message. The hybrid-DE continues using the full-DE until the KL divergence is smaller than a predefined target KL divergence or a predetermined maximum number of full-DE iterations is reached when it then switches to a Gaussian approximation DE. Thus, we can trade-off for efficiency of the hybrid-DE method by varying the target KL divergence and/or the maximum number of full-DE iterations.
VI. IMPLICATION OF GAUSSIAN APPROXIMATIONS FOR CODE DESIGN

A. Threshold comparison of density evolution using Gaussian approximations

Table I gives the number of floating point operations per edge per decoding iteration for each of the DE algorithms. We do not show the overhead operations (such as computing the KL divergence) that do not occur during the DE iterations in Table I. We have found that these operations make only a small contribution to the overall overhead. The relative complexity and accuracy of each approach will depend on the size of the lookup table chosen (for Approximations 1 and 3) and the number of quantization points chosen to sample the PDF (for full-DE) or for Q-function evaluation (for Approximation 2).

In Figs. 6 to 8, we compare the percentage of threshold error and CPU time gain, with respect to the threshold and CPU time obtained from full-DE with 10000 decoding iterations, for MET-LDPC codes of different rates. We select these MET-LDPC code structures (code A-G in Table IV in Appendix) such that they contain degree-one and punctured variable nodes in order to emphasize the benefits of hybrid-DE over Gaussian approximations. We specified 98000 quantization points per PDF for full-DE and lookup table sizes of 10001 and 38302 for Approximations 1 and 3 respectively. This is because we found that assigning smaller lookup tables (for Approximations 1 and 3) and smaller number of quantization points (for full-DE) reduces the accuracy of the threshold calculation.

Figs. 6 and 7 present the effect of the maximum number of full-DE iterations and the target KL divergence on the accuracy of the threshold calculation in hybrid-DE, respectively. We compare the percentage of threshold error of hybrid-DE, with respect to the threshold obtained from full-DE with 1000 full-DE iterations, for MET-LDPC codes in Table IV. It is clear from Figs. 6 and 7 that we can trade-off the accuracy of threshold calculation by varying maximum number of full-DE iterations and target KL divergence accordingly. For the purpose of comparison, variation of the percentage of threshold error of full-DE with a set of maximum number of full-DE iterations is also shown in the Fig. 6. It is clear from Fig. 6 that even when we limit the number of full-DE iterations in hybrid-DE algorithm, there is still considerable performance improvement in terms of threshold accuracy to be gained by continuing with Gaussian approximation iterations compared to the full-DE threshold with the same maximum number of full-DE iterations but without the additional Gaussian approximation iterations.

In Figs. 8(a) and 8(b), we respectively compare the percentage of threshold error and CPU time gain, with respect to the threshold and CPU time obtained from full-DE with 1000 decoding iterations with the three single-parameter Gaussian approximation methods, and with the hybrid-DE. We calculate the threshold using hybrid-DE method for a range of target KL divergences and maximum number of full-DE iterations in order to emphasize the trade-off between accuracy and efficiency. It can be seen from Fig. 8 that we can obtain up to 10 times computational time gain by doing hybrid-DE 2 and 3, with only loosing maximum of 5% accuracy of the threshold calculation. However, even though the all Gaussian approximation methods report a better CPU time gain than hybrid-DE, they accurately estimate the code threshold only at higher rates, i.e., Approximation 1 and 3 estimate the code threshold with less than 5% error only for code rates above 0.6 where as Approximation 2 gives an accurate estimations only at rates above 0.7. Furthermore it is clear from Fig. 8 that, even by doing only 10 full-DE iterations in hybrid-DE (hybrid-DE 1), we can still get a considerable accuracy improvement of threshold calculation compared to the single-parameter Gaussian approximation methods. These make hybrid-DE more suitable for code design where accurate and efficient threshold calculation is particularly valuable.

B. Design of MET-LDPC codes

The aim of this section is to show how approximate DE algorithms affect the design of optimal MET-LDPC code ensembles (defined by the degree distribution with the largest possible code threshold). This is a non-linear cost function maximization problem, where the cost function is the code threshold and the degree distributions are the variables to be optimized. It is still possible to obtain an optimal degree distribution even if the DE approximation returns an inaccurate threshold, as long as it returns the highest threshold for the optimal degree distributions. However this is not the case using Gaussian approximations. For example, we perform an exhaustive search on a single parameter ($\alpha_3$) of a MET-LDPC

| FP operation | MET-DE | App. 1 (Mean) | App. 2 (BER) | App. 3 (RCA) | Hybrid-DE$^4$ |
|-------------|--------|---------------|--------------|--------------|---------------|
| Sums        | VN     | CN            | VN           | CN           | VN            | CN            |
| Multiplications | VN | CN            | VN           | CN           | VN            | CN            |
| Lookup-tables | VN | d_{c,1}       | d_{e,1}      | d_{e,1}      | VN            | CN            |
| Exponentials | VN     | d_{e,1}       | d_{e,1}      | d_{e,1}      | VN            | CN            |
| Q-functions | VN     | d_{e,1}       | d_{e,1}      | d_{e,1}      | VN            | CN            |
| Convolutions | VN     | d_{c,1}       | d_{e,1}      | d_{e,1}      | VN            | CN            |

$^4\alpha$ is the percentage of MET-DE iterations.

TABLE I
FLOATING POINT (FP) OPERATIONS PER EDGE, BASED ON AN AVERAGE EDGE DEGREE OF VARIABLE NODE (VN), $d_e$ AND AN AVERAGE EDGE DEGREE OF CHECK NODE (CN), $d_c$
Fig. 6. Percentage of threshold error, with respect to the full-DE threshold with 1000 decoding iterations, for different maximum number of full-DE iterations. After the maximum number of full-DE iterations is reached, full-DE stops, while hybrid-DE continues with a Gaussian approximation (Approximation 1) for up to 1000 decoding iterations. No KL divergence limit is set for hybrid-DE.

Fig. 7. Percentage of threshold error, with respect to the full-DE threshold with 1000 decoding iterations, for different target KL divergence limits. No maximum number of full-DE iterations is set. The hybrid-DE algorithm swaps to Gaussian iterations (Approximation 1) only once the target KL divergence is met.

code with the remaining parameters fixed in Fig. 9. The maxima of the full-DE does not coincide with the maxima of the approximations. While the hybrid-DE cost function closely follows the shape of the full-DE cost function the other approximations do not. This threshold difference between full-DE and Gaussian approximations can significantly impact the search for good code ensembles for given design constraints.

To further demonstrate the effect of the Gaussian approximations on code design, we design rate 1/10 and 1/2 MET-LDPC codes on B1-AWGN channel with full-DE, hybrid-DE and the three Gaussian approximations stated in Section III. We use the joint optimization methodology proposed in [20] to design MET-LDPC codes. The results are presented in Tables II and III where the values are rounded off to four decimal places. For a fair comparison, we consider similar MET-LDPC code structures from Table X and VI of [4] for rate 1/10 and 1/2 MET-LDPC codes respectively. The maximum number of decoding iterations and target bit error

Threshold error $= \left| 1 - \frac{\sigma^*_{DE}}{\sigma^*_{App}} \right|$, where, $\sigma^*_{App}$ is the threshold calculated using relevant method and $\sigma^*_{DE}$ is the threshold calculated using full-DE with the full 1000 iterations.

CPU time gain $= \frac{CPU_{SimFullDE}}{CPU_{SimApp}}$ Algorithms were written in Matlab and run on an Intel Xeon E5-2650, 2.6 GHz PC. The maximum number of decoding iterations were the same for all the MET-LDPC codes considered.

Fig. 8. (a) Percentage of threshold error, with respect to the full-DE threshold with 1000 decoding iterations. (b) CPU time gain for one DE calculation, with respect to the full-DE with 1000 decoding iterations, when channel noise standard deviation is 0.01 below the code threshold. Rates 0.1 to 0.7 correspond to codes A to G in Table VII in Appendix.

Fig. 9. Threshold found through exhaustive search for a rate 1/2 MET-LDPC code with $L(r, x) = a_1 r_1 x_1^2 + a_2 r_2 x_2^3 + a_3 r_3 x_3^3 + a_4 r_1 x_4$ for $a_1 = 0.5$, $a_3 = a_4$ and $a_2 = (1 - a_1 - a_3)$. The rate for the BP decoding process is set to 1000 and $10^{-10}$ respectively. For hybrid-DE, we set the target KL divergence to 0.04 and the maximum number of full-DE iterations allowed to 100 and calculate KL divergence after every 5 decoding iterations to check whether the message PDFs are close to Gaussian. The results in Table II show that Approximations 1 and 2 can result in noticeable inaccuracy for designing rate 1/10 MET-LDPC codes. However, the rate 1/10 MET-LDPC code designed using hybrid-DE closely matches the MET-LDPC code designed with full-DE. The results in Table III show that the Approximations 1 and 2 are more successful at designing rate 1/2 MET-LDPC codes and the worst performing algorithm in this case was Approximation 3. Nevertheless, the rate 1/2 MET-LDPC code designed with hybrid-DE gives the closest match to the MET-LDPC code designed with full-DE.

We then simulate the finite-length performances for the rate 1/10 MET-LDPC codes with degree distributions from Table II with block length of 100000. As expected, the threshold differences between the ensembles shown in Table II are reflected in the finite-length performance differences in Fig. 10.

This suggests that our proposed hybrid-DE method performs similarly to full-DE, making it suitable for code optimization even at low rate and with punctured variable nodes. Since our
TABLE II
OPTIMIZATION OF RATE 1/10 MET-LDPC CODES ON BI-AWGN CHANNEL

| Design with (Table X of [3]) | MET-LDPC code | Threshold |
|-------------------------------|----------------|-----------|
| Reference code                | \( L(r, x) = 0.1r_1 x_1^2 x_2^2 + 0.025r_1 x_1 x_2 + 0.875r_1 x_3 \) | \( R(x) = 0.025x_1 x_2 + 0.875x_3 x_4 \) | \( \sigma_{app} \) | \( \sigma_{opt} \) |
|                              | \( L(r, x) = 0.0775r_1 x_1 x_2 x_3 + 0.0477r_1 x_1 x_2 x_4 + 0.8747r_1 x_4 \) | \( R(x) = 0.0014x_1 x_2 + 0.0028x_1 x_3 + 0.0214x_2 x_3 + 0.0414x_2 x_4 + 0.8335x_3 x_4 \) | - | 2.5346 |
| Full-DE                       | \( L(r, x) = 0.0538r_1 x_1 x_2 x_3 + 0.0775r_1 x_1 x_2 x_4 + 0.8687r_1 x_4 \) | \( R(x) = 0.0116x_1 x_2 + 0.0137x_1 x_3 + 0.0061x_2 x_3 + 0.0573x_2 x_4 + 0.8113x_3 x_4 \) | 2.5455 | 2.5372 |
| Hybrid-DE                     | \( L(r, x) = 0.0544r_1 x_1 x_2 x_3 + 0.0641r_1 x_1 x_2 x_4 + 0.8815r_1 x_4 \) | \( R(x) = 0.0099x_1 x_2 + 0.0035x_1 x_3 + 0.0051x_2 x_3 + 0.8355x_2 x_4 + 0.0460x_3 x_4 \) | 2.4661 | 2.4965 |
| Approximation 1 (Mean)        | \( L(r, x) = 0.06r_1 x_1 x_2 x_3 + 0.0576r_1 x_1 x_2 x_4 + 0.8824r_1 x_4 \) | \( R(x) = 0.0058x_1 x_2 + 0.0118x_1 x_3 + 0.1833x_2 x_3 + 0.6991x_3 x_4 \) | 2.3659 | 2.3850 |
| Approximation 2 (BER)         | \( L(r, x) = 0.0942r_1 x_1 x_2 x_3 + 0.0336r_1 x_1 x_2 x_4 + 0.8722r_1 x_4 \) | \( R(x) = 0.0006x_1 x_2 + 0.0107x_1 x_3 + 0.0165x_2 x_3 + 0.0262x_2 x_4 + 0.8459x_3 x_4 \) | 2.5056 | 2.5303 |

TABLE III
OPTIMIZATION OF RATE 1/2 MET-LDPC CODES ON BI-AWGN CHANNEL

| Design with (Table VI of [3]) | MET-LDPC code | Threshold |
|-------------------------------|----------------|-----------|
| Reference code                | \( L(r, x) = 0.2r_0 x_1 x_2 x_3 + 0.5r_1 x_1^2 + 0.3r_1 x_2 + 0.2r_1 x_4 \) | \( R(x) = 0.1x_1 x_2 + 0.4x_1 x_3 + 0.4x_2 x_4 \) | \( \sigma_{app} \) | \( \sigma_{opt} \) |
|                              | \( L(r, x) = 0.4162r_0 x_1 x_2 x_3 + 0.5629r_1 x_1^2 + 0.0294r_1 x_2 + 0.4076r_1 x_4 \) | \( R(x) = 0.1848x_1 x_2 + 0.2191x_1 x_3 + 0.1047x_2 x_3 + 0.3905x_2 x_4 + 0.0171x_3 x_4 \) | - | 0.9656 |
| Full-DE                       | \( L(r, x) = 0.5962r_0 x_1 x_2 x_3 + 0.0004r_1 x_1^2 x_2 x_3 + 0.1055r_1 x_1 x_2 + 0.8941r_1 x_4 \) | \( R(x) = 0.0189x_1 x_2 + 0.0679x_1 x_3 + 0.1153x_2 x_3 + 0.8935x_2 x_4 + 0.0006x_3 x_4 \) | 0.9660 | 0.9688 |
| Hybrid-DE                     | \( L(r, x) = 0.2792r_0 x_1 x_2 x_3 + 0.4066r_1 x_1^2 + 0.2341r_1 x_1 x_2 + 0.3592r_1 x_4 \) | \( R(x) = 0.0024r_0 x_1 x_2 + 0.1618r_1 x_1^2 x_2 + 0.2558x_1 x_3 + 0.2401x_2 x_3 + 0.1191x_3 x_4 \) | 0.9152 | 0.9588 |
| Approximation 1 (Mean)        | \( L(r, x) = 0.5034r_0 x_1 x_2 x_3 + 0.0068r_1 x_1^2 x_2 x_3 + 0.2337r_1 x_1 x_3 + 0.7595r_1 x_4 \) | \( R(x) = 0.0016r_0 x_1 x_2 + 0.2201r_1 x_1^2 x_2 + 0.0223x_1 x_3 + 0.0018x_2 x_3 + 0.7576x_3 x_4 \) | 0.9099 | 0.9535 |
| Approximation 2 (BER)         | \( L(r, x) = 0.1564r_0 x_1 x_2 x_3 + 0.3689r_1 x_1^2 + 0.4607r_1 x_1 x_2 + 0.1704r_1 x_4 \) | \( R(x) = 0.0168x_1 x_2 + 0.2934x_1 x_3 + 0.1758x_2 x_3 + 0.0419x_2 x_4 + 0.1285x_3 x_4 \) | 0.9435 | 0.9420 |

The proposed method can also be used to strike a balance between efficiency and accuracy required, we claim that the proposed hybrid method is a suitable DE approximation technique for code design.

VII. CONCLUSION

This paper investigated the performance of density evolution for low-density parity-check (LDPC) and multi-edge type low-density parity-check (MET-LDPC) codes over the binary input additive white Gaussian noise channel. We applied and analyzed three single-parameter Gaussian approximation models. We showed that the accuracy of single-parameter Gaussian approximations might be poor under several conditions, namely codes at low rates and codes with punctured variable nodes. Then, we proposed a more accurate density evolution (DE) approximation, referred to as hybrid-DE, which is a combination of the full-DE and a single-parameter Gaussian approximation. With hybrid-DE, we avoided the symmetric Gaussian assumption at early decoding iterations of BP decoding, making our code threshold calculations significantly more accurate than existing methods of using Gaussian approximations for all decoding iterations. At the same time, hybrid-DE significantly reduced the computational time of evaluating the code threshold compared to full-DE. These make hybrid-DE more suitable for the code design where accurate and efficient threshold calculation is particularly valuable. Finally, we considered code optimization and presented a code design by using full-DE, hybrid-DE and three Gaussian approximations. The designed codes using hybrid-DE closely match with the codes designed using full-DE. Thus, we can suggest that the hybrid-DE is a good DE technique for code design. Since hybrid-DE is not specific to MET-LDPC codes, it also can be used for designing other codes defined on graphs such as irregular LDPC codes.
Fig. 10. The bit error rate performance of length-100000, rate 1/10 MET-LDPC codes with degree distributions from Table IV on BI-AWGN channel.

APPENDIX

TABLE IV
MET-LPDC CODES USED IN FIGS. 6 TO 8

| Code | Rate | Degree distribution |
|------|------|---------------------|
| Code A | 0.1 | \[L(x, r) = 0.6737r_1 x_1^2 + 0.3263 r_1 x_1^3 + 0.0001 r_0 x_1^2 x_2 + 0.0001 r_1 x_4 \]
| | | \[R(x) = 0.3737 x_1^2 + 0.5260 x_1^2 + 0.0003 x_1 x_2 + 0.0001 x_2 x_4 \]
| Code B | 0.2 | \[L(x, r) = 0.7281 r_1 x_1^2 + 0.0052 r_1 x_1^3 + 0.2669 r_0 x_1^2 x_2 + 0.2669 r_1 x_4 \]
| | | \[R(x) = 0.1284 x_1 x_2 + 0.6711 x_1 x_2 + 0.0006 x_1 x_2 + 0.2669 x_1 x_4 \]
| Code C | 0.3 | \[L(x, r) = 0.7213 r_1 x_1^2 + 0.0006 r_1 x_1^3 + 0.2781 r_0 x_1^2 x_2 + 0.2781 r_1 x_4 \]
| | | \[R(x) = 0.5656 x_1 x_2 + 0.097 x_1 x_2 + 0.044 x_1 x_2 + 0.2781 x_1 x_4 \]
| Code D | 0.4 | \[L(x, r) = 0.6864 r_1 x_1^2 + 0.0007 r_1 x_1^3 + 0.3129 r_0 x_1^2 x_2 + 0.3129 r_1 x_4 \]
| | | \[R(x) = 0.2613 x_1 x_2 + 0.1638 x_1 x_2 + 0.1749 x_1 x_2 + 0.3129 x_1 x_4 \]
| Code E | 0.5 | \[L(x, r) = 0.5713 r_1 x_1^2 + 0.1788 r_1 x_1^3 + 0.2497 r_0 x_1^2 x_2 + 0.2497 r_1 x_4 \]
| | | \[R(x) = 0.2507 x_1 x_2 + 0.0699 x_1 x_2 + 0.1793 x_1 x_2 + 0.2497 x_1 x_4 \]
| Code F | 0.6 | \[L(x, r) = 0.5001 r_1 x_1^2 + 0.3 r_1 x_1^3 + 0.1999 r_0 x_1^2 x_2 + 0.1999 r_1 x_4 \]
| | | \[R(x) = 0.0998 x_1 x_2 + 0.1205 x_1 x_2 + 0.1197 x_1 x_2 + 0.1197 x_1 x_4 \]
| Code G | 0.7 | \[L(x, r) = 0.3501 r_1 x_1^2 + 0.6190 r_1 x_1^3 + 0.0309 r_0 x_1^2 x_2 + 0.0309 r_1 x_4 \]
| | | \[R(x) = 0.1428 x_1^2 + 0.0645 x_1^2 + 0.0927 x_1 x_2 + 0.0309 x_1 x_4 \]

REFERENCES

[1] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, “Design of capacity-approaching irregular low-density parity-check codes,” IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[2] R. M. Tanner, “A recursive approach to low complexity codes,” IEEE Trans. Inform. Theory, vol. 27, no. 5, pp. 533–547, 1981.

[3] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, “Improved low-density parity-check codes using irregular graphs,” IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 585–598, 2001.

[4] T. J. Richardson and R. L. Urbanke, “Multi-edge type LDPC codes,” in Workshop honoring Prof. Bob McEliece on his 60th birthday, California Institute of Technology, Pasadena, California, 2002.

[5] T. J. Richardson and R. Urbanke, Modern coding theory. Cambridge University Press, 2008.

[6] S.-Y. Chung, T. J. Richardson, and R. L. Urbanke, “Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation,” IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 657–670, 2001.

[7] S.-Y. Chung, “On the construction of some capacity-approaching coding schemes,” Ph.D. dissertation, MIT, Cambridge, MA, 2000.

[8] F. Lehmann and G. M. Maggio, “Analysis of the iterative decoding of LDPC and product codes using the Gaussian approximation,” IEEE Trans. Inform. Theory, vol. 49, no. 11, pp. 2993–3000, 2003.

[9] S. Ten Brink, “Convergence behavior of iteratively decoded parallel concatenated codes,” IEEE Trans. Commun., vol. 49, no. 10, pp. 1727–1737, 2001.

[10] M. Fu, “On Gaussian approximation for density evolution of low-density parity-check codes,” in Proc. IEEE Int. Conf. Commun., vol. 3, 2006, pp. 1107–1112.

[11] M. Ardakani and F. R. Kschischang, “A more accurate one-dimensional analysis and design of irregular LDPC codes,” IEEE Trans. Commun., vol. 52, no. 12, pp. 2106–2114, 2004.

[12] K. Xie and J. Li, “On accuracy of Gaussian assumption in iterative analysis for LDPC codes,” in Proc. IEEE Int. Symp. Inf. Theory, 2006, pp. 2398–2402.

[13] L. Schmalen and S. T. Brink, “Combining spatially coupled LDPC codes with modulation and detection,” in Proc. 9th Int. ITG Conf. on Systems, Comm. and Coding (SCC). VDE, 2013, pp. 1–6.

[14] W. Ryan and S. Lin, Channel codes: classical and modern. Cambridge University Press, 2009.

[15] M. K. Simon, Probability distributions involving Gaussian random variables: A handbook for engineers and scientists. Springer Science & Business Media, 2007.

[16] A. Araujo and E. Giné, The central limit theorem for real and Banach valued random variables. Wiley New York, 1980, vol. 431.

[17] N. C. Beaulieu and Q. Xie, “An optimal lognormal approximation to lognormal sum distributions,” IEEE Trans. Veh. Technol., vol. 53, no. 2, pp. 479–489, 2004.

[18] T. M. Cover and J. A. Thomas, Elements of information theory. John Wiley & Sons, 2012.

[19] T. K. Moon, Error Correction Coding : Mathematical Methods and Algorithms. John Wiley and Son, 2005.

[20] S. Jayasooriya, S. J. Johnson, L. Ong, and R. Berretta, “Optimization of graph based codes for belief propagation decoding,” in Proc. IEEE Inf. Theory Workshop, 2014, pp. 456–460.