The Three/Two Gaussian Parametric LDLC Lattice Decoding Algorithm and Its Analysis

Ricardo Antonio Parrao Hernandez *Student Member* and Brian M. Kurkoski *Member*, IEEE

***Abstract*—Low density lattice codes (LDLC) are high dimensional lattices with a sparse inverse generator matrix, that can be decoded efficiently using iterative decoding. In the iterative LDLC decoder the messages are Gaussian mixtures, and for any implementation, the Gaussian mixtures must be approximated. This work describes a parametric LDLC decoding algorithm, where internally at the variable node infinite Gaussian mixtures are approximated with three or two Gaussians, while the messages between nodes are single Gaussians. Strengths of the algorithm include its simplicity and suitability for analysis. Analysis is performed by evaluating the Kullback-Leibler divergence between the true messages and the three/two Gaussian approximation. The approximation using three or two Gaussians is more accurate than previously proposed approximations. Also, noise thresholds for the proposed LDLC decoder are presented, the proposed decoder reduces the noise thresholds 0.05 dB compared to previous parametric decoders. The numerical results show that for $n = 100$ and $n = 1,000$ the two-Gaussian approximation is the same as the full-complexity decoder. But when the dimension is $n = 10,000$, a three-Gaussian approximation is needed.**

***Index Terms*—low-density lattice codes, parametric decoder, lattice decoder**

I. Introduction

Lattice codes are codes over the real numbers which possess great potential to become an efficient, practical and reliable communication scheme for the AWGN channel. Shannon showed that codes with very long random Gaussian-distributed codewords can approach the AWGN capacity [1], and now it is known that lattice codes can also achieve the AWGN capacity [2–4]. Lattices are especially appealing for multi-terminal Gaussian networks, where the encoder and the channel use the same real algebra.

Low density lattice codes (LDLC), introduced by Sommer, Feder and Shalvi [5], are high-dimensional lattices defined by a sparse inverse generator matrix. The construction and decoding of LDLCs resemble low density parity check codes (LDPC), that is, using a belief propagation (BP) decoding algorithm on a sparse graph. It was reported that the LDLC belief propagation decoder attains a symbol error rate of 10^{-5} at 0.6 dB from the unconstrained AWGN channel capacity. Already, relaying and physical layer network coding schemes that use LDLCs have been described [6–10].

In the LDLC belief propagation decoder the messages passed between check and variable nodes are continuous functions. In any implementation, these continuous functions must be approximated. In the original implementation [5], these messages were approximated by a discretely quantized function. The amount of quantization, typically 1024 bins, is impracticably large.

A computationally efficient approach is to represent the messages as a mixture of Gaussian functions. For the AWGN channel, the messages are precisely represented using a mixture containing an infinite number of Gaussians. This is also impractical, so it is natural to approximate these messages with a finite mixture of Gaussians.

Parametric LDLC decoding algorithms employ Gaussian mixtures to approximate the messages. An LDLC decoder using a Gaussian mixture reduction algorithm was introduced in [11]. All possible pairs of Gaussians on a list are searched and the closest pairs are replaced with a single Gaussian. Further, using single Gaussians as the messages between the variable and check nodes leads to reduced memory requirements with a minor performance penalty [12]. Yona and Feder [13] presented a parametric decoder, where the Gaussian mixture approximation is made by taking the dominating Gaussian in the mixture. This process is done by searching in tables, which are sorted in terms of the mixture coefficients. But these relatively complicated operations, whether the Gaussian mixture reduction or sorting, must be performed at every multiplication at the variable node, on each iteration. Such operations may not be suitable if LDLC decoders are to be implemented in hardware. In [14] a single-Gaussian moment matching (SGMM) approximation was used internally at the variable node for every incoming message, and density evolution noise thresholds were presented. The SGMM is computationally efficient but finite-dimensional results were not given.

This paper presents a parametric decoding algorithm for LDLC lattices. In the proposed algorithm the infinite Gaussian mixtures are approximated with three or two Gaussians, which are nearby to the channel message. Accordingly, we call this the “three/two Gaussian parametric LDLC decoder”.

The major advantage of the proposed LDLC decoding algorithm is a favorable performance-complexity tradeoff as compared to previous parametric decoding algorithms such as the GMR algorithm [11] and the table search algorithm [13]. For small to medium-dimension lattices of $n \leq 1000$, the tradeoff is particularly favorable using $M = 2$ Gaussians in the approximation, we show performance practically indistinguishable from the GMR algorithm, but complexity significantly lower. For higher-dimensional lattices of $n = 10000$, the two-Gaussian approximation has some performance loss, which is recovered by increasing to an $M = 3$ Gaussian approximation.

Ricardo Antonio Parrao Hernandez and Brian M. Kurkoski are with School of Information Science at the Japan Advanced Institute of Science and Technology, Ishikawa, Japan e-mail: ricardo.parrao@jaist.ac.jp, kurkoski@jaist.ac.jp. Part of this work appeared in the proceedings of the IEEE Iformation Theory Workshop (ITW 2015), Jeju, Korea.This work was supported by JSPS Kakenhi Grant Number 26289119

Another advantage of the proposed algorithm is that it is nearly parameter-free; the only parameter selection of interest is using $M = 2$ or $M = 3$ Gaussians. This is in contrast to other LDLC decoders that have algorithmic parameters, e.g. threshold parameters and list size of the GMR algorithm. In addition, the approximation used at the variable node is particularly susceptible to numerical analysis. We show the accuracy of the approximation by evaluating the Kullback-Leibler (KL) divergence. This gives some insight into the performance-complexity tradeoff for LDLC decoding, and shows that the three/two Gaussian approximation is more accurate than the SGMM used in [14].

In addition, this paper evaluates the noise thresholds for the three/two Gaussian parametric decoding algorithm, and compares them to those of the single Gaussian decoding [14]. The proposed decoder reduces the noise threshold 0.05 dB for node degree $d = 7$. The best-known LDLC noise thresholds were given in [15] for spatially-coupled LDLC lattices but finite-length results were not given. Also, the noise thresholds of the "dithered" LDLC construction [5] are found.

The structure of the paper is as follows. Section II gives a definition of lattices and describes low density lattice codes. Section III describes the operations over Gaussian mixtures and the moment matching approximation. Section IV gives the three/two Gaussian approximation algorithm and its analysis using KL divergence. Section V describes the three/two Gaussian parametric decoding algorithm, its complexity and convergence properties. Section VI shows the numerical results for different lattice dimensions, and finally, Section VII summarizes the paper.

II. Lattices and Low Density Lattice Codes

A lattice Λ is an additive subgroup of $\mathbb{R}^n$. A matrix $\mathbf{G}$, whose columns are the basis vectors, is called the generator matrix of the lattice. A lattice point is defined as:

$$\mathbf{x} = \mathbf{G}\mathbf{b} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{b} \in \mathbb{Z}^n$ are column vectors.

The Voronoi region of a lattice is defined as the set of all points that are closer to a lattice point than to any other. The volume of the Voronoi region is given by $|\det(\mathbf{G})|$.

A lattice codeword $\mathbf{x}$ is transmitted over the AWGN channel. Then it is received as:

$$\mathbf{y} = \mathbf{x} + \mathbf{z}, \tag{2}$$

where the vector $\mathbf{z}$ is additive Gaussian noise with 0 mean and variance σ^2, $z_i \sim \mathcal{N}(0, \sigma^2)$ for $i = 1, 2 \ldots, n$.

A maximum-likelihood decoder estimates $\hat{\mathbf{x}}$ as the received codeword:

$$\hat{\mathbf{x}} = \underset{\mathbf{x} \in \Lambda}{\operatorname{argmax}} \Pr(\mathbf{y}|\mathbf{x}). \tag{3}$$

If $\mathbf{x} = \hat{\mathbf{x}}$ the correct codeword is received, or an error occurred otherwise.

Because we consider only the unconstrained lattice transmission, this paper considers the volume-to-noise ratio (VNR) as an analog to signal-to-noise ratio. The VNR is defined as:

$$VNR = \frac{|\det(\mathbf{G})|^{2/n}}{2\pi e \sigma^2}. \tag{4}$$

The lattice capacity is when $\sigma^2 = \frac{1}{2\pi e}$ [16], which corresponds to $VNR = 0$dB.

A. Low Density Lattice Codes

A low density lattice code (LDLC), introduced by Sommer et al. [5], is an n-dimensional lattice code defined by a non-singular generator matrix $\mathbf{G}$ satisfying the property that the inverse generator matrix $\mathbf{H} = \mathbf{G}^{-1}$ is sparse.

We consider LDLC lattices with regular latin square matrix $\mathbf{H}$ with row/column degree d. We choose the values for the d non-zero elements, called the generator sequence $\mathbf{h}$, to satisfy $h_1 \geq h_2 \geq \cdots \geq h_d > 0$. The signs of the generator sequence entries in the inverse generator matrix $\mathbf{H}$ are randomly changed to " $-$ " with probability one-half.

A necessary condition to achieve exponential convergence of the message variance is to select the generator sequence such that:

$$\alpha = \frac{\sum_{i=2}^{d} h_i^2}{h_1^2} < 1 \tag{5}$$

holds [5].

B. LDLC Decoder

Similar to low density parity check (LDPC) codes, a bipartite graph is defined with variable nodes corresponding to a single element of the lattice codeword $\mathbf{x} = \mathbf{G}\mathbf{b}$ and check nodes corresponding to a check equation of the form $\sum_k h_k x_{i_k}$ is an *integer*, where i_k are the positions of the non-zero elements at the corresponding row of the inverse generator $\mathbf{H}$, and the *integer* is unknown. An edge connects variable node i and check node j if and only if $\mathbf{H}_{i,j} \neq 0$.

In [5] an iterative decoder for LDLCs was presented. The iterative decoder passes messages over the bipartite graph, and the messages between variable and check nodes are real functions. For the AWGN channel these functions are Gaussian mixtures. For implementation these Gaussian mixture needs to be quantized, accordingly we call it "quantized" algorithm and is as follows:

- *Initialization*: Each variable node k sends the function $f_k(w)$ to all d connected check nodes, which is a single Gaussian given by:

$$f_k(w) = \mathcal{N}(w; y_k, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(w-y_k)^2}{2\sigma^2}}, \tag{6}$$

  for $k = 1, 2 \ldots, n$, where $\mathbf{y} = \mathbf{G}\mathbf{b} + \mathbf{z}$ is the channel message and $\mathbf{z}$ is the additive Gaussian noise with 0 mean and variance σ^2.
- *Check-to-variable node message*: On edge j the check-to-variable node messages $R_j(w)$ are calculated as follows:
  1) Convolution: All messages except $f_j(w)$, are convolved after expanding each message by its coeffi-

cient h_j

$$\tilde{p}_j(w) = f_1(\frac{w}{h_1}) * \cdots * f_{j-1}(\frac{w}{h_{j-1}}) * f_{j+1}(\frac{w}{h_{j+1}}) * \dots f_d(\frac{w}{h_d}). \quad (7)$$

for $j = 1, 2 \dots d$.

2) Stretching and periodic extension: The convolution $\tilde{p}_j(w)$ is stretched by $-h_j$, and extended to a periodic function with period $1/|h_j|$:

$$p_j(w) = \tilde{p}(-h_j w). \quad (8)$$

$$R_j(w) = \sum_{i=-\infty}^{\infty} p_j(w - \frac{i}{h_j}). \quad (9)$$

- *Variable-to-check node*: On edge j the variable-to-check node messages $f_j(w)$ are calculated in two steps:
  1) Product : The product of all incoming messages, except for the message j, is performed:

$$\tilde{f}_j(w) = \mathcal{N}(w; y_k, \sigma^2) \prod_{\substack{l=1,\\ l\neq j}}^{d} R_l(w). \quad (10)$$

  2) Normalization: $\tilde{f}_j(w)$ is normalized as:

$$f_j(w) = \frac{\tilde{f}_j(w)}{\int_{-\infty}^{\infty} \tilde{f}_j(w) dw}. \quad (11)$$

These steps are repeated until the maximum number of iteration is reached.

- *Final decision*: At the last iteration the product without omitting any message is done:

$$\tilde{f}_k^{final}(w) = \mathcal{N}(w; y_k, \sigma^2) \prod_{l=1}^{d} R_l(w). \quad (12)$$

And the estimate the lattice point $\hat{\mathbf{x}}$ and the integer information $\hat{\mathbf{b}}$ are

$$\hat{x}_k = \underset{w}{\operatorname{argmax}} \tilde{f}_k^{final}(w) \quad (13)$$

$$\hat{\mathbf{b}} = \lfloor H\hat{\mathbf{x}} \rceil. \quad (14)$$

respectively.

The quantized algorithm requires high memory and high computation time. Also, a poor choice of the quantization resolution can produces errors, i.e. a zero output instead of an impulse.

III. Operations on Gaussian Mixtures

This section describes the product of Gaussian mixtures and the moment matching approximation.

Let $f(w)$ be a mixture of N Gaussians,

$$f(w) = \sum_{i=1}^{N} c_i \mathcal{N}(w; m_i, v_i), \quad (15)$$

with mean m_i, variance v_i and mixing coefficients $c_i > 0$ and $\sum_{i=1}^{N} c_i = 1$ for $i = 1, 2 \dots, N$.

A. Product over Gaussian mixtures

The product of two Gaussian mixtures $f(w) = \sum_{i=1}^{N} f_i(w)$ and $g(w) = \sum_{j=1}^{M} g_j(w)$ is $f(w) \cdot g(w)$. The product of two components $f_i(w) = c_1\mathcal{N}(w; m_1, v_1)$ and $g_j(w) = c_2\mathcal{N}(w; m_2, v_2)$ is a single Gaussian $s(w) = c\mathcal{N}(w; m, v)$ with mean m, variance v and mixing coefficient c given by:

$$\frac{1}{v} = \frac{1}{v_1} + \frac{1}{v_2} \quad (16)$$

$$\frac{m}{v} = \frac{m_1}{v_1} + \frac{m_2}{v_2} \quad (17)$$

$$c = \frac{c_1 c_2}{\sqrt{2\pi(v_1 + v_2)}} e^{-\frac{(m_1 - m_2)^2}{2v_1 + 2v_2}}. \quad (18)$$

The Gaussian product $f(w) \cdot g(w)$ is the mixture of the $N \cdot M$ products obtained using the pair-wise operation above.

B. Moment Matching Approximation

The "moment matching approximation", is the single-Gaussian approximation of a Gaussian mixture $f(w)$, given by (15), with a single Gaussian $q(w) = \mathcal{N}(w; m, v)$ which minimizes the Kullback-Leiber divergence between $f(w)$ and $q(w)$ [17, appdx. A]. The moment-matching approximation (MM) finds the single Gaussian $q(w)$ which has the same mean m and variance v as $f(w)$. The mean m and variance v is given by:

$$m = \sum_{i=1}^{N} c_i m_i \quad (19)$$

$$v = \sum_{i=1}^{N} c_i m_i^2 - (\sum_{i=1}^{N} c_i m_i)^2. \quad (20)$$

This operation is denoted as:

$$q(w) = \mathrm{MM}(f(w)). \quad (21)$$

IV. Three/Two Gaussian approximation

In this section we describe an approximation of the product of a single Gaussian and an infinite Gaussian mixture, which is key for understanding the behavior of the three/two Gaussian parametric decoding algorithm. The exact product is also infinite, and the motivation is to select those few Gaussians which are dominant. By evaluation of the Kullback-Leiber divergence, we show that selecting two or three Gaussians gives good approximations.

Having a good approximation at the variable node is a key step for accurate performance in the parametric LDLC decoder. We have observed that the approximation in the tails is very important. A poor approximation in the Gaussian messages causes errors to accumulate as the LDLC iterative decoding progresses.

Instead of calculating the periodic expansion over all integers, is convenient to use a reduced number of integers. Since the periodic expansion takes place at the variable node, and due to multiplication with the channel message, the resultant periodic Gaussians that are far from the channel message have near-zero mixing coefficients.

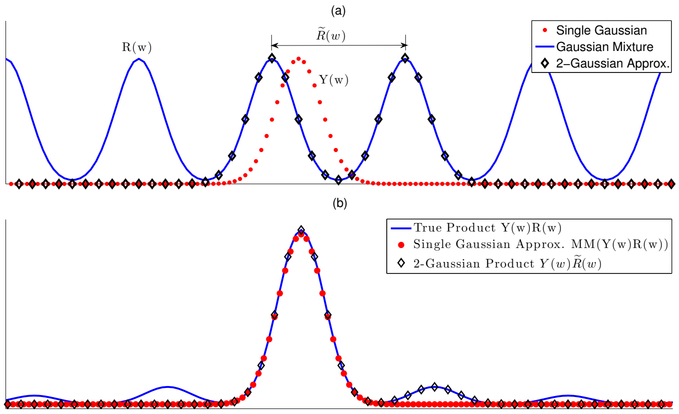

Fig. 1: Multiplication of a Gaussian mixture and a single Gaussian, and the approximation with a Ssngle Gaussian. This operation take place at the variable node. (a) shows the individual mixtures, (b) the true product and the approximations.

The single Gaussian $Y(w)$ represents the channel message and has mean m_a and variance v_a with $v_a = \sigma^2$. The Gaussian mixture $R(w)$ represents the incoming check-to-variable node message and is a periodic mixture of Gaussians with period $\frac{1}{|h|}$ for $h \in \mathbf{h}$, and parameters m_c and v_c, and $R(w)$ is given by:

$$R(w) = \sum_{i=-\infty}^{\infty} \mathcal{N}(w; m_c + \frac{i}{h}, v_c). \tag{22}$$

And let $\tilde{R}(w)$ be the summation in (22) restricted to some finite integer set $\mathcal{B}$. We want to approximate an infinite Gaussian mixture $Y(w)R(w)$ with $Y(w)\tilde{R}(w)$, which consists of a finite number of Gaussians. In Fig. 1-(a) $Y(w)$ and $R(w)$ are illustrated. In Fig. 1-(b) the true product $Y(w)R(w)$ and the single Gaussian moment matching approximation $MM\big(Y(w)R(w)\big)$ are shown. The true product and the MM approximation are visually similar, but with a closer look at the tails, a difference exists. The MM is a poor approximation of $Y(w)R(w)$, and empirically causes problems with iterative decoding.

A. Gaussian Neighbors Selection

Here we consider the two cases of $|\mathcal{B}| = 3$ and $|\mathcal{B}| = 2$ Gaussians neighbors near m_a. Let the two-Gaussian set be $\mathcal{B} = \{b_1, b_2\}$ with $b_2 = b_1 + 1$, and the three-Gaussian set be $\mathcal{B} = \{b_0, b_1, b_2\}$, with $b_0 = b_1 - 1$ and $b_2 = b_1 + 1$.

For the two-Gaussian set we select two integers, one less than, and one greater than a non-integer estimate. Find b_1 such that:

$$\frac{b_1}{h} + m_c < m_a < \frac{b_1 + 1}{h} + m_c, \tag{23}$$

for $h > 0$. That is,

$$b_1 = \lfloor -h(m_c - m_a) \rfloor. \tag{24}$$

And in the three-Gaussian set we choose the nearest Gaussian and its two nearest neighbors. That is:

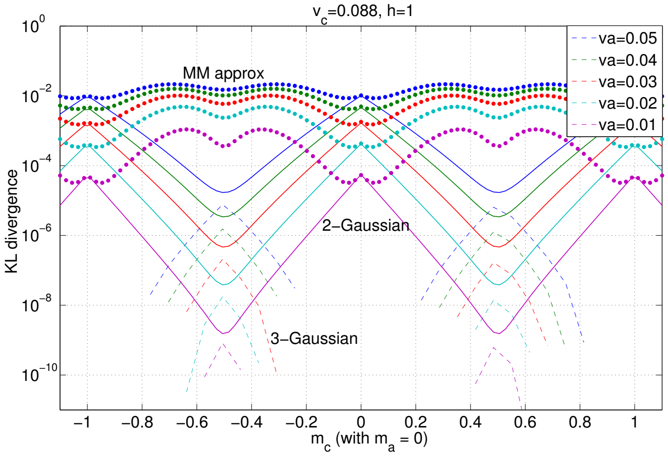

Fig. 2: KL divergence for the dominant message ($h = 1$), for single Gaussian approximation (dotted line), two-Gaussian approximation (solid line) and three-Gaussian approximation (dash line). For $v_c = 0.088$, which corresponds to an early iteration.

$$b_0 = b_1 - 1, \tag{25}$$

$$b_1 = \lceil h(m_c - m_a) \rfloor, \text{ and} \tag{26}$$

$$b_2 = b_1 + 1. \tag{27}$$

The resulting mixture is:

$$\tilde{R}(w) = \mathcal{N}(w; \frac{b_1}{h} + m_c, v_c) + \mathcal{N}(w; \frac{b_2}{h} + m_c, v_c), \tag{28}$$

for the two-Gaussian set. And for three-Gaussian set it is:

$$\tilde{R}(w) = \mathcal{N}(w; \frac{b_0}{h} + m_c, v_c) + \mathcal{N}(w; \frac{b_1}{h} + m_c, v_c) + \mathcal{N}(w; \frac{b_2}{h} + m_c, v_c), \tag{29}$$

where $\tilde{R}(w)$ is the approximation of $R(w)$.

B. Kullback-Leiber divergence

In this section a heuristic analysis of the approximation accuracy is shown, by evaluating the Kullback-Leiber divergence. Clearly, the accuracy of the approximation depends on the number of Gaussians taken. By selecting a small number of Gaussians we want to minimize the KL divergence between $Y(w)R(w)$ and the approximation $Y(w)\tilde{R}(w)$. The KL divergence is given by:

$$\int_{-\infty}^{\infty} Y(w)R(w) \log \frac{Y(w)R(w)}{Y(w)\tilde{R}(w)} dw. \tag{30}$$

When $Y(w)R(w)$ and $Y(w)\tilde{R}(w)$ are a mixture of Gaussians, selecting mean and variance close to each other will minimize the divergence. While the KL divergence between two Gaussian mixtures is not analytically tractable in general, various approximations for the KL divergence in general Gaussian mixture models were proposed [18]. However, these approximations are not suitable for the Gaussian mixtures

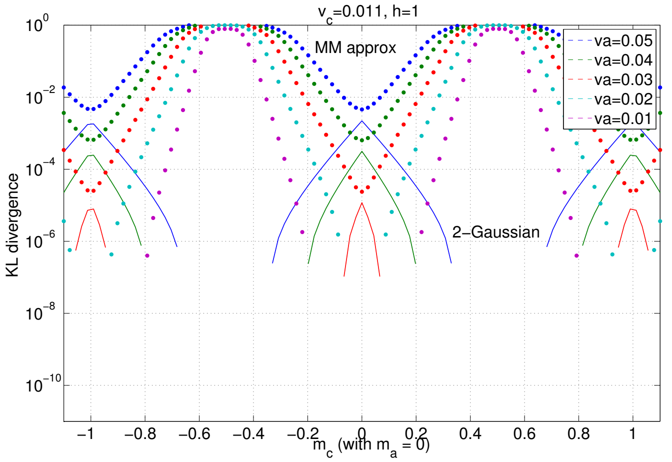

Fig. 3: KL divergence for the dominant message ($h = 1$), for single Gaussian approximation (dotted line), two-Gaussian approximation (solid line). For $v_c = 0.011$, which corresponds to an intermediate iteration, and the single Gaussian is not accurate.

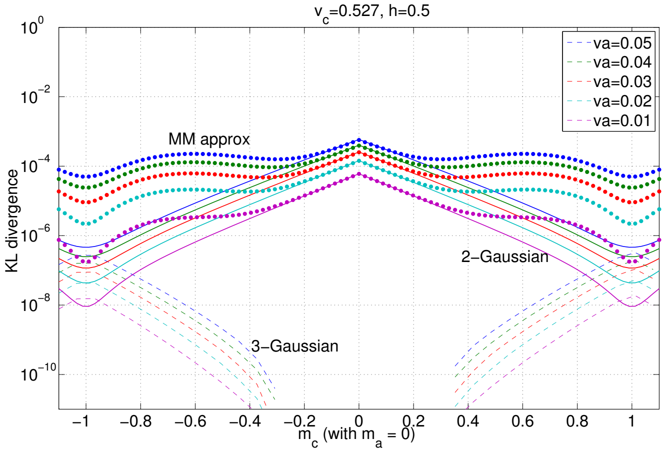

Fig. 4: KL divergence for the non-dominant message ($h_i < 1$), for single Gaussian approximation (dotted line), two-Gaussian approximation (solid line) and three-Gaussian approximation (dash line), for $v_c = 0.527$

which occur during the message passing decoding of lattices, in the sense they do not give insight to the problem. Instead the KL divergence is evaluated numerically, this has the advantage of giving the exact value.

Figures 2–4 show the KL divergence for the single-Gaussian moment matching approximation (dotted line), three-Gaussian approximation (dash line) and two-Gaussian approximation (solid line), using typically observed values for v_a and v_c under LDLC decoding. We present all values for m_c, but not all are equally likely because m_c is not uniformly distributed. The Kullback-Leiber divergence only depends on $m_c - m_a$, so we set $m_a = 0$. The worst case is when $h = 1$.

Fig. 2 shows $v_c = 0.088$, corresponding to an early decoding iteration. Here, even the MM approximation has a KL divergence of less than 10^{-2}. Empirically, we have observed that a KL divergence of greater than 10^{-2} is a poor approximation for the proposed LDLC decoding algorithm. But KL divergence of less than 10^{-3} at least gives visually similar Gaussian functions.

Fig. 3 shows $v_c = 0.011$, corresponding to intermediate iterations of LDLC decoding, where the MM presents worse KL divergence. The KL divergence for two-Gaussian approximation is always less than 10^{-2} and the three-Gaussian approximation is even better. This suggests that the two-Gaussian approximation may be sufficient. The simulation results will show that this is often true, but when the dimension is very large, the three-Gaussian approximation is more reliable.

In Fig. 4 the MM presents a good approximation for the non-dominant edge ($h_i < 1$). The message on the edge with the highest value in the generator sequence ($h_i = 1$; dominant edge) gives more reliable information during the message passing. For this reason the dominant edge required more accurate approximation.

In order to have a low complexity parametric decoding algorithm a single Gaussian approximation is desired. However, using a single Gaussian approximation is not accurate. As is shown in Fig. 1, the single Gaussian approximation presents a large difference in the tails of the Gaussian distribution which contribute significantly in the Kullback-Leiber divergence. In the moment matching algorithm the tails are very important to obtain a good approximation. For this reason the single Gaussian approximation is not suitable for a good approximation.

Note that the approximation given in this section does not minimize the Kullback-Leiber divergence, but is used because it is efficient for a decoder to implement. By maintaining the dominant Gaussians in the mixture, the approximation is a good one, as we have shown in this section.

V. Three/Two Gaussian Parametric Decoder

The proposed parametric LDLC decoding algorithm is presented here. Internally at the variable node, messages are represented by mixtures of Gaussians, but externally only single Gaussians are used. There are two types of approximations used internally at the variable node: (1) the 3/2 Gaussian approximation from Sec. IV-A, and (2) the SGMM approximation used before variable node output; previous work [12] has shown that a single Gaussian message from the variable node to the check node is sufficient.

The variable-to-check message along edge k is a single Gaussian denoted $f_k(w)$. The check-to-variable message along edge k is a single Gaussian denoted $R_k(w)$. Single Gaussians are represented by its mean and variance. Thus, storage of variable-to-check messages requires $2 \cdot n \cdot d$ elements, and likewise for the check-to-variable messages.

A. Three/Two Gaussian Parametric Decoder Description

For the AWGN channel, let the received message be

$$y_k(w) = \mathcal{N}(w; y_k, \sigma^2). \tag{31}$$

- *Check Node*: The incoming messages are d single Gaussians $f_i(w) = \mathcal{N}(w; m_i, v_i)$. The output message $\tilde{p}_i(w)$

at the convolution step is a single Gaussian with mean $\tilde{m}_i$ and variance $\tilde{v}_i$ given by:

$$\tilde{m}_i = -\frac{1}{h_i}\sum_{\substack{j=1\\j\neq i}}^{d} h_j m_j \tag{32}$$

$$\tilde{v}_i = \frac{1}{h_i^2}\sum_{\substack{j=1\\j\neq i}}^{d} h_j^2 v_j. \tag{33}$$

The computation of $\tilde{m}$ and $\tilde{v}$ can be performed using a forward-backward recursion.

- *Variable node*: The messages coming from the check node are single Gaussian $\mathcal{N}(w;\tilde{m}_i,\tilde{v}_i)$. Then the expansion step (periodic with period $1/|h_i|$ if $\mathcal{B}=\mathbb{Z}$) is approximated by:

$$\tilde{R}_i(w) = \sum_{b\in\mathcal{B}} \mathcal{N}(w; m_i(b), \tilde{v}_i) \tag{34}$$

where the mean m of each Gaussian is

$$m_i(b) = \tilde{m}_i + \frac{b}{h_i}, \tag{35}$$

and the set $\mathcal{B}$ represents a subset of the integers, e.g. two or three. Rather than searching over all integers $\mathcal{B}=\mathbb{Z}$, instead select two or three integers close to the channel message, as described in previous section. The message $f_i(w)$ sent back to the check node is a single Gaussian approximated by:

$$p_i(w) = y_k(w)\prod_{\substack{j=1\\j\neq i}}^{d} \tilde{R}_j(w), \tag{36}$$

$$f_i(w) = \mathrm{MM}\Big(p_i(w)\Big), \tag{37}$$

where $y_k(w) = \mathcal{N}(w; y_k, \sigma^2)$ is the channel message, and $\tilde{R}_j(w)$ is the approximation of the periodic expansion. To maintain low complexity, just a single Gaussian is used in the messages between variable and check nodes. The single Gaussian message to send to the check node is calculated by the moment matching algorithm.

A forward-backward recursion can be used to reduce the number of operations to calculate the variable-to-check node messages, as shown in the next section.

B. Forward-backward recursion

Computing the output $f_i(w)$ at the variable node k can be implemented by a forward-backward recursion. This recursion is distinct from previously described forward-backward approaches [11] in how the channel value y_k is handled — in the three/two Gaussian parametric decoding algorithm the channel message is multiplied last (although the channel message y_k is used to select the periodic Gaussians).

The forward-backward recursion is done as follows:

1) The forward recursion defined as:

$$\alpha_j(w) = \alpha_{j-1}(w)\cdot\tilde{R}_j(w), \tag{38}$$

for $j = 2, 3, \ldots, d-1$, with $\alpha_1(w)$ initialized as equal to $\tilde{R}_1(w)$.

2) The backward recursion $\beta_j(w)$ is computed, for $j = d-2, d-3, \ldots, 1$, as:

$$\beta_{j-1}(w) = \beta_j(w)\cdot\tilde{R}_{j-1}(w), \tag{39}$$

with $\beta_{d-1}(w)$ initialized as the approximation $\tilde{R}_d(w)$.

3) Then combining the forward and backward recursion, we get:

$$\tilde{f}_i(w) = \alpha_{i-1}(w)\cdot\beta_i(w). \tag{40}$$

4) Finally, the single Gaussian output of the variable node is calculated by using the moment matching approximation:

$$f_i(w) = \mathrm{MM}\Big(y_k(w)\cdot\tilde{f}_i(w)\Big). \tag{41}$$

C. Three/Two Gaussian Parametric Decoder Complexity

The complexity of the three/two Gaussian parametric decoding algorithm is dominated by the forward and backward algorithm which is $\mathcal{O}(n{\cdot}t{\cdot}2^{d-1})$ if messages are approximated by two Gaussians, and $\mathcal{O}(n\cdot t\cdot 3^{d-1})$ if three Gaussians are selected, where t is the number of iterations, n is the lattice dimension and d is the degree of the LDLC inverse generator matrix.

For comparison, the complexity of the quantized BP decoding algorithm [5] is $\mathcal{O}(n\cdot t\cdot d\cdot\frac{L}{\Delta})$ where Δ is the probability density function resolution and L is the range length, and is dominated by a discrete Fourier transform. The complexity for [11] is $\mathcal{O}(n\cdot d\cdot t\cdot K^2\cdot M^4)$, and is dominated by a moment matching algorithm. And for [13] the complexity is $\mathcal{O}(n{\cdot}d{\cdot}t{\cdot}K{\cdot}M^3)$, and is dominated by sorting and searching in tables, where K is the number of replications and M the number of Gaussian used in the mixtures. The common values which present similar performance are $n=100$, $d=5$, $L=4$, $\Delta=\frac{1}{256}$, $M=6$ and $K=3$. The parametric decoder presented in [13] requires storing an $n\cdot d$ list of M Gaussians.

In addition to the asymptotic complexity, we also compare the computation time of the three/two Gaussian parametric decoding algorithm with that of the GMR algorithm. The performance of the GMR decoder presented in [11] depends on two parameters, the Gaussian quadratic loss threshold θ and the maximum number of Gaussians M. A small value of θ presents a better approximation (and thus better performance) but the computational complexity increases. In Fig. 5 a computation time comparison for one iteration between the GMR decoder and the three/two Gaussian parametric decoding algorithm is shown. We simulated 1000 codewords for dimension $n = 1000$, at $VNR = 2$dB, and found the average time for one iteration. The proposed decoding algorithm presents a lower computation time for the values of θ for which GMR decoding [11] has similar performance measured using symbol-error rate (see Sec. VI) . The three/two Gaussian parametric decoding algorithm is independent of the value of θ. In fact, the only parameter is whether there are two or three Gaussians.

In Fig. 6, the average number of iterations required for decoder convergence is shown. We took a sample of 1000

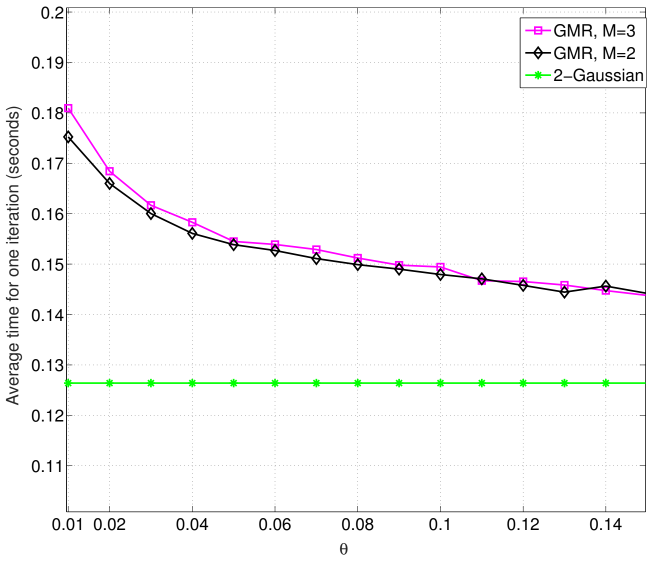

Fig. 5: Time comparison between GMR algorithm [11], with $M = 2$ and $M = 3$, and the three/two Gaussian parametric decoding algorithm, for lattice dimension $n = 1000$ and $VNR = 2$.

converged codewords (non-converging cases are ignored) and evaluated the mean of the number of iterations needed. The average number of iteration reduces as VNR increases. The use of three Gaussians does not reduce the average number of iterations. The GMR decoder for $M = 2$ and $\theta = 0.5$ required more iterations on average to converge. The three/two Gaussian parametric decoding algorithm has lower complexity, since the average number of iterations to converge is fewer, and the time consumed for one iteration is constant for any fixed VNR. As far as we are aware, convergence for LDLCs has not been presented in the literature before.

The storage needed for the three/two Gaussian parametric decoding algorithm is $2 \cdot n \cdot d$. Because the message passed between check and variable nodes are single Gaussians, these messages are parameterized by two values, the mean and the variance. Internally in the variable node the temporary storage needed is 2^{d-1} and 3^{d-1}, for the two-Gaussian and three-Gaussian approximation respectively. The use of a larger number of Gaussians as an approximation, increment the complexity, where the storage required internally at the variable node is 4^{d-1} or more.

D. Three/Two Gaussian Parametric Decoding Algorithm

The proposed decoding algorithm is as follows:

Input: The received message $\mathbf{y} = \mathbf{G}\mathbf{b} + \mathbf{z}$, the channel variance σ^2, the inverse generator $\mathbf{H}$ and the maximum number of iterations $iter\_max$.

Output: The estimated information $\hat{\mathbf{b}}$.

1) At variable node k, for $k = 1, 2, \ldots, n$, send to all connected check nodes the message y_k and σ^2.
2) At check node k every message $\tilde{p}_i(w)$ is a single Gaussian, for $i = 1, 2, \ldots, d$. The mean and variance are computed as in equation (32) and (33).

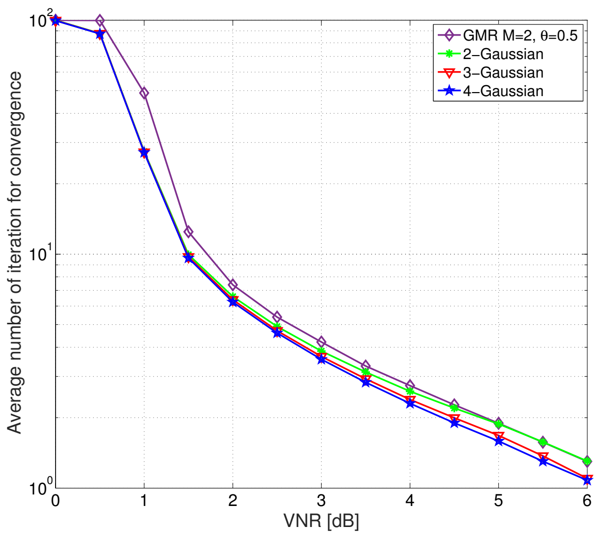

Fig. 6: Average number of iterations required for decoder convergence in terms of the VNR, for LDLC dimension $n = 1000$ and degree $d = 7$.

3) At the variable node k the $\tilde{R}_i(w)$ message, for $i = 1, 2, \ldots, d$, is calculated by selecting 2 or 3 Gaussians as describe in section IV-A.
4) The selecting mixtures are multiplied except the message i, to calculate
$$\prod_{\substack{j=1 \\ j \neq i}}^{d} \tilde{R}_j(w). \tag{42}$$
5) Then $p(w)$is calculated by multiplying the channel message $y_k(w)$ as in equation (36).
6) A moment matching is performed to send back to the check node a single Gaussian message $f_i(w)$.
7) Steps 2-6 are repeated until the maximum number of iterations $iter\_max$ is reached.
8) The final estimate $\hat{x}_k$ is made by combining all messages at the variable node, and $\hat{x}_k$ is the mean of
$$\mathrm{MM}\Big(y_k(w) \prod_{i=1}^{d} \tilde{R}_i(w)\Big). \tag{43}$$
9) Finally the received message is estimated by
$$\hat{\mathbf{b}} = \mathbf{H}\hat{\mathbf{x}}. \tag{44}$$

VI. Numerical Results

A. Noise Thresholds

Noise thresholds are used to evaluate the performance of the three/two Gaussian parametric decoding algorithm. The noise threshold is the lowest VNR for which three/two Gaussian parametric decoding of asymptotically large-dimensional lattice converges. Performing exact density evolution would require the joint distribution for the mean and variance of the messages sent between the variable node and check node, which are the parameters used for the messages in the

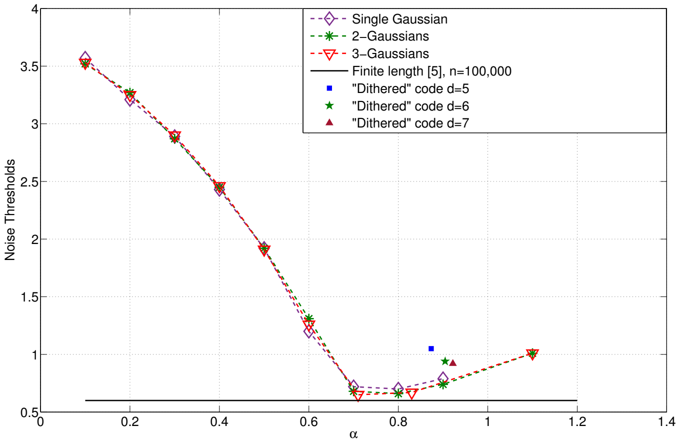

Fig. 7: Noise thresholds, measured in distance from capacity, for three/two Gaussian decoder and the single Gaussian decoder [14], for various LDLC lattices with parameters $d = 7$ and α.

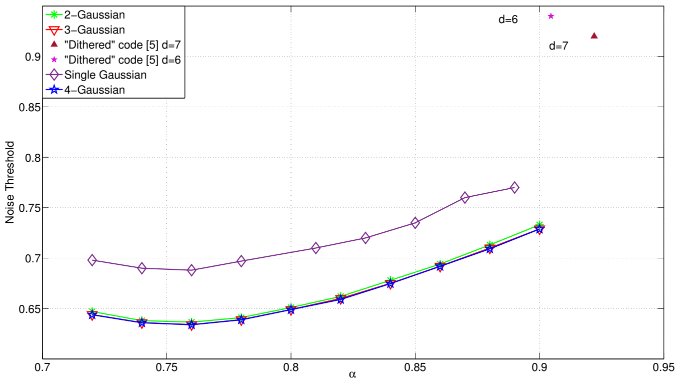

Fig. 8: Noise thresholds details, measured in distance from capacity, for the three/two Gaussian parametric decoder with different values of $M = \{2, 3, 4\}$ and the single Gaussian decoder [14], for various LDLC lattices with parameters $d = 7$ and α.

decoding algorithm. The evaluation of exact density evolution for two variables is computationally demanding. Instead we perform Monte Carlo density evolution which has been used for non-binary low density parity check codes [19].

The evaluation of Monte Carlo density evolution is similar to the one given in [14], and is as follows. We consider a lattice construction with generator sequence $\mathbf{h} = \{1, w, \ldots, w\}$, where w is given by:

$$w = \sqrt{\frac{\alpha}{d-1}}, \tag{45}$$

and α is defined in (5), a necessary condition for exponential convergence of the belief-propagation decoder [5]. We generate a data pool, consisting of two types of elements: ones with label 1 denoted by $P_{(1)}$ and others with label w denoted by $P_{(w)}$. $P_{(1)}$ consists in $N_{(1)} = 10^6$ messages and $P_{(w)}$ consists of $N_{(w)} = (d-1) \cdot 10^5$ messages, i.e.

$$P_{(1)} = \{(m_1, v_1), \ldots, (m_{N_{(1)}}, v_{N_{(1)}})\} \tag{46}$$

$$P_{(w)} = \{(m_1, v_1), \ldots, (m_{N_{(w)}}, {}_{N_{(w)}})\}, \tag{47}$$

where m and v denote the mean and variance respectively.

The messages (m_l, v_l), for all $l = 1, \ldots, N_{(1)}$ ($l = 1, \ldots, N_{(w)}$ for the w label message), are initialized as follows. The noise variance σ^2 is assigned to v_l, and m_l is initialized with the received symbol generated from $\mathcal{N}(0, \sigma^2)$, since the all zero codeword (lattice point) is assumed.

At each half iteration the variable/check node input consists of one element of $P_{(1)}$ and $d-1$ elements from $P_{(w)}$. The check and variable nodes are computed as shown in Section V-A, and stored in an output pool. The output pool becomes the input pool for the next half iteration. The mean of the variable-to-check node messages for the w labeled edge was used to check for convergence. When the mean of all $v_i \in P_{(w)}$ samples fell below to 0.001, within 50 iterations, then convergence was declared.

The noise thresholds, obtained using the three/two Gaussian parametric decoding algorithm and the single-Gaussian decoder [14], are shown in Fig. 7 for several values of α and $d = 7$. In addition, the noise thresholds for the sequence $\{\frac{1}{2.31}, \frac{1}{3.17}, \frac{1}{5.11}, \frac{1}{7.33}, \frac{1}{11.71}, \frac{1}{13.11}, \frac{1}{17.55}\}$, and further normalized with $\frac{1}{2.31}$ to obtain $\mathbf{h} = \{1, 0.73, 0.45, 0.32, 0.20.18, 0.13\}$ for $d = 5$, $d = 6$ and $d = 7$ as proposed in [5] are shown; this LDLC is denoted as the "dithered" code. In Fig. 8 a closer look at the noise thresholds, shows that using $M = 4$ Gaussians for the approximation does not significantly reduce the noise threshold compared to $M = 3$ Gaussians. The noise thresholds for the three/two parametric LDLC decoder are reduced by 0.05 dB compared with the noise thresholds for the single-Gaussian decoder in [14]. Interestingly, the noise thresholds for the considered lattice construction are slightly better than the dithered code.

B. Finite-length results

To evaluate the three/two Gaussian parametric LDLC decoder the all zeros codeword was simulated over the AWGN channel. The inverse generator matrix was generated as in [5], with the generator sequence $\mathbf{h} = \{1, \frac{1}{\sqrt{d}}, \ldots, \frac{1}{\sqrt{d}}\}$. With this choice of generator sequence we have $\alpha = \frac{d-1}{d}$. The inverse generator was further normalized in order to have $\sqrt[n]{|\det(\mathbf{H})|} = 1$.

Different lattice dimensions $n = 100$, $n = 1000$ and $n = 10000$ were simulated, and the inverse generator $\mathbf{H}$ has degree $d = 3$ for dimension $n = 100$, and $d = 7$ for dimension $n = 1000$ and $n = 10000$. The symbol error rate (SER; a symbol error is $\hat{b}_k \neq b_k$) vs. the VNR was evaluated.

In the three/two parametric decoder algorithm three cases are simulated. The first and second case are when the approximation is done by two and three Gaussians, and are denoted by "2-Gaussian" and "3-Gaussian" respectively. Ideally the message variance approaches 0 at the estimated lattice point. This behavior is dominated by the message with the highest generator sequence value ($h_i = 1$) called "the dominant message". The third case is approximating the dominant message by three Gaussians and the non-dominant messages ($h_i < 1$) are approximated with two Gaussians.

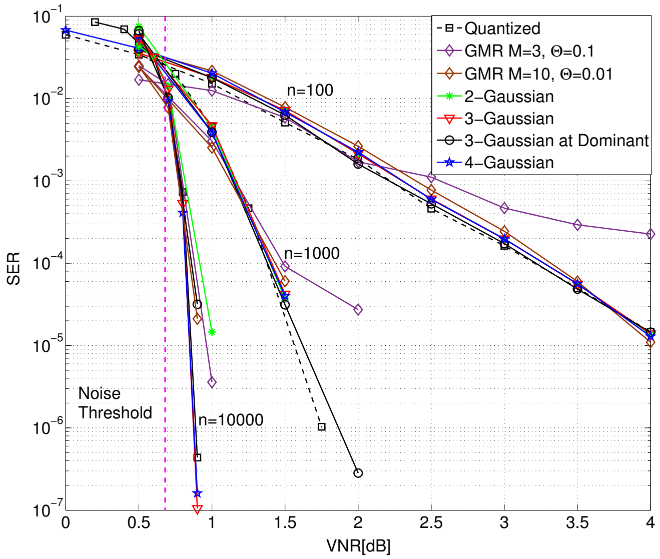

Fig. 9: SER vs the gap from capacity for dimension $n = 100$, $n = 1000$, $n = 10000$

As shown in Fig. 9 the three/two Gaussian parametric decoding algorithm performs nearly as well as the quantized algorithm [5] in SER, especially for dimension $n = 1,000$. For $n = 10,000$ approximation with three Gaussians are needed for similar performance to the quantized algorithm. In addition a comparison using 4-Gaussians was evaluated. Even if the approximation is performed using $M = 4$ Gaussians, which implies a reduction in the KL divergence, this did not give any visible improvement, due to the discrete nature of the decoder errors.

Also a comparison with the GMR decoder [11] with $M = 3$ and $\theta = 0.1$, similar parameters in terms of number of Gaussians in the mixtures. The proposed decoding algorithm has a better performance compared to the GMR algorithm with similar parameters, and has lower complexity. The GMR decoder with $\theta = 0.01$ and $M = 10$, high complexity parameters, performs nearly similar to the proposed decoder.

The constructed LDLC for $d = 7$ has $\alpha = 0.8571$ and its noise threshold is 0.68 as shown in Fig. 9. The gap to the noise threshold for the three/two Gaussian parametric decoding algorithm is 0.22dB at SER of 10^{-7} when the approximation is done with three Gaussians for lattice dimension $n = 10,000$.

VII. Conclusion

In this work we presented the three/two Gaussian parametric decoding algorithm for low density lattice codes (LDLC), which is a reliable and efficient decoding algorithm. In addition the proposed decoding algorithm maintains low complexity. This is because the messages between variable and check nodes are only single Gaussian functions.

The three/two Gaussian parametric decoding algorithm approximates the Gaussian mixture with only two or three Gaussians which are close to the channel value. These selections are more accurate approximations than the single Gaussian approximation, in terms of the Kullback-Leiber divergence. The advantages of the three/two Gaussian parametric decoding algorithm are:

1) Reduces the noise thresholds compared to single Gaussian decoders
2) Has low complexity not only in terms in computation time per iteration but also in the average number of iterations to converge. Convergence in terms of volume-to-noise ratio for LDLC is presented for the first time in the literature. The noise threshold analysis and the average number of iteration presented in this work are a guideline for LDLC lattice design.
3) Has symbol error rate nearly similar compared to the best-known decoding algorithm.
4) Has only one parameter that is the number of Gaussians (three or two) needed for approximating the messages.

These characteristics makes the three-two parametric decoder algorithm attractive for different applications, such as those where a compute-and-forward scheme [6] are used and/or crypto-systems which use LDLC lattices [20].

References

[1] C. E. Shannon, "Probability of error for optimal codes in a Gaussian channel," *Bell System Technical Journal, The*, vol. 38, pp. 611–656, 1959.

[2] R. de Buda, "The upper error bound of a new near-optimal code," *Info. Theory, IEEE Trans. on*, vol. 21, pp. 441–445, Jul. 1975.

[3] U. Erez and R. Zamir, "Achieving $\frac{1}{2}\log(1 + \text{SNR})$ on the AWGN channel with lattice encoding and decoding," *Info. Theory, IEEE Trans. on*, vol. 50, no. 10, pp. 2293–2314, Oct. 2004.

[4] T. Linder, C. Schlegel, and K. Zeger, "Corrected proof of de Buda's theorem," *Info. Theory, IEEE Trans. on*, vol. 39, no. 5, pp. 1735–1737, Sep 1993.

[5] N. Sommer, M. Feder, and O. Shalvi, "Low-density lattice codes," *Information Theory, IEEE Transactions on*, vol. 54, no. 4, pp. 1561–1585, April 2008.

[6] B. Nazer and M. Gastpar, "Compute-and-forward: Harnessing interference through structured codes," *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 6463–6486, Oct 2011.

[7] B. Chen, D. Jayakody, and M. Flanagan, "Low-density lattice coded relaying with joint iterative decoding," *Communications, IEEE Transactions on*, vol. 63, no. 12, pp. 4824–4837, Dec 2015.

[8] Y. Wang and A. Burr, "Physical-layer network coding via low density lattice codes," in *Networks and Communications (EuCNC), 2014 European Conference on*, June 2014, pp. 1–5.

[9] N. Ferdinand, M. Nokleby, and B. Aazhang, "Low density lattice codes for the relay channel," in *Communications (ICC), 2013 IEEE International Conference on*, June 2013, pp. 3035–3040.

[10] B. Chen, D. N. K. Jayakody, and M. F. Flanagan, "Distributed low-density lattice codes," *IEEE Communications Letters*, vol. 20, no. 1, pp. 77–80, Jan 2016.

[11] B. Kurkoski and J. Dauwels, "Message-passing decoding of lattices using Gaussian mixtures," in *Info.Theory, 2008. ISIT 2008. IEEE International Symposium on*, July 2008, pp. 2489–2493.

[12] ——, "Reduced-memory decoding of low-density lattice codes," *Communications Letters, IEEE*, vol. 14, no. 7, pp. 659–661, July 2010.

[13] Y. Yona and M. Feder, "Efficient parametric decoder of low density lattice codes," in *Information Theory, 2009. ISIT 2009. IEEE International Symposium on.* Seoul, Korea: IEEE, Jun.-Jul. 2009, pp. 744–748.

[14] B. Kurkoski, K. Yamaguchi, and K. Kobayashi, "Single-Gaussian messages and noise thresholds for decoding low-density lattice codes," in *Information Theory, 2009. ISIT 2009. IEEE International Symposium on.* Seoul, Korea: IEEE, Jun.–Jul. 2009, pp. 734–738.

[15] H. Uchikawa, B. Kurkoski, K. Kasai, and K. Sakaniwa, "Threshold improvement of low-density lattice codes via spatial coupling," in *Computing, Networking and Communications (ICNC), 2012 International Conference on*, Jan 2012, pp. 1036–1040.

[16] G. Poltyrev, "On coding without restrictions for the AWGN channel," *Info. Theory, IEEE Trans. on*, vol. 40, no. 2, pp. 409–417, Mar. 1994.

[17] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2005.

[18] J. R. Hershey and P. A. Olsen, “Approximating the kullback leibler divergence between gaussian mixture models,” in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4, April 2007, pp. IV–317–IV–320.

[19] M. C. Davey, “Error-correction using low-density parity-check codes,” Ph.D. dissertation, University of Cambridge, 1999.

[20] R. Hooshmand, T. Eghlidos, and M. R. Aref, “Improving GGH public key scheme using low density lattice codes,” *CoRR*, vol. abs/1503.03292, 2015. [Online]. Available: http://arxiv.org/abs/1503.03292