Code-Aware Power Allocation for Irregular LDPC Codes

Zeina Mheich and Valentin Savin(✉)

CEA-LETI, MINATEC Campus, 38054 Grenoble, France
{Zeina.Mheich,Valentin.Savin}@cea.fr

Abstract. In this paper, we investigate a code-dependent unequal power allocation method for Gaussian channels using irregular LDPC codes. This method allocates the power for each set of coded bits depending on the degree of their equivalent variable nodes. We propose a new algorithm to optimize the power allocation vector using density evolution algorithm under the Gaussian approximation. We show that unequal power allocation can bring noticeable gains on the threshold of some irregular LDPC codes with respect to the classical equal power allocation method depending on the code and the maximum number of decoding iterations.

Keywords: Irregular LDPC codes · Unequal power allocation · Density evolution · Gaussian approximation

1 Introduction

The Shannon capacity of the power constrained point-to-point Gaussian channel is achieved using independent and identically distributed (i.i.d.) symbols. Therefore, unequal power allocation (UPA) does not increase capacity. However, Shannon did not provide any practical coding/decoding scheme to achieve this capacity. Nowadays, there exists powerful codes approaching this capacity as LDPC codes. It was shown in [1] that irregular LDPC codes perform better than regular ones in terms of threshold. In irregular LDPC codes, the variable nodes do not have the same degree and thus are not equally protected. Thus we expect that the performance of the code will be affected if the power allocated for some set of symbols associated to variable nodes of a certain degree is different from that allocated for a set of symbols associated to variable nodes of an another degree. Therefore, it is not known if equal power allocation (EPA) is also optimal for practical irregular LDPC codes. This will be the subject of investigation in our work. In reference [2], the authors investigate the UPA problem for irregular LDPC codes for point-to-point Gaussian channels. They obtained up to 0.25 dB of gain with respect to EPA method. However, the authors use in [2] Monte-Carlo simulations in order to optimize the power allocation vector, which

This work was carried out within the framework of Celtic-Plus SHARING project.

is time-consuming. Usually, the LDPC code is chosen to have a good threshold assuming that the decoder can perform unlimited number of iterations. In this work, we consider that a given irregular LDPC code is used at the transmitter and we optimize the power allocation for a target bit error rate and using a decoder with limited number of iterations. We propose a new algorithm to optimize the power allocation vector such that the noise threshold of the existing irregular LDPC code family is maximized. Hence, we propose a modified density evolution algorithm using Gaussian approximation [3] when the UPA method is used at the transmitter. We apply this algorithm for the Gaussian point-to-point channel and the Gaussian relay channel.

2 LDPC Codes and Density Evolution

This section recalls some of the basics about LDPC codes. We consider the point-to-point Gaussian channel case. At the source side, a message of k information bits is encoded by a LDPC encoder to a n-bit codeword. When the transmitter uses an equal power allocation strategy, the coded bits are modulated using a BPSK constellation, such that the bits 0 and 1 are mapped into $+1$ and -1 respectively. The resulting sequence after modulation is denoted by $x^n = [x_1, \cdots, x_n]$. The symbols $x_i,\ i \in \{1, \cdots, n\}$ are transmitted over a discrete-time memoryless additive white Gaussian noise channel. The channel output corresponding to the input x is $y = x + z$, where z is a noise following a normal distribution of zero mean and of variance σ^2. Consider an LDPC code characterized by a Tanner graph $\mathcal{H}$, with n variable nodes and m check nodes ($m = n - k$). An irregular LDPC code is characterized by bit nodes and check nodes with varying degrees. The fraction of edges which are connected to degree-i variable nodes is denoted λ_i, and the fraction of edges which are connected to degree-i check nodes, is denoted ρ_i. The functions $\lambda(x) = \sum_{i=2}^{d_v} \lambda_i x^{i-1}$ and $\rho(x) = \sum_{i=2}^{d_c} \rho_i x^{i-1}$ are defined to describe the degree distributions from the perspective of Tanner graph edges. By definition $\lambda(1) = 1$ and $\rho(1) = 1$. An alternative characterization of the degree distribution for the variable nodes $\Lambda(x) = \sum_{i=2}^{d_v} \Lambda_i x^i$, from the perspective of Tanner graph nodes, will be used also in this paper. Hence, Λ_i designates the fraction of degree-i variable nodes. The decoding algorithms used to decode LDPC codes are collectively called message-passing algorithms since they operate by the passing of messages along the edges of a Tanner graph. Under a message-passing algorithm, variable nodes and check nodes exchanges messages iteratively. Each node processes the received messages on the edges connected to it and sends messages back to its neighbors such that the output message is a function of all incoming messages to the node except the incoming message on the edge where the output message will be sent. The sum-product decoding is a message-passing algorithm in which the messages are log likelihood ratios (LLR) and the calculations at the variable and check nodes are performed using sum and product operations. Hence, a message can be written as the LLR of the equally probable random variable $x \in \{+1, -1\}$:

$$\mathrm{LLR} = \log \frac{p(x = +1|w)}{p(x = -1|w)}, \tag{1}$$

where w is a random variable describing all the information incorporated into this message. The sum-product algorithm iteratively computes an approximation of the maximum a posteriori (MAP value) for each code bit. However, the a posteriori probabilities returned by the sum-product decoder are only exact MAP probabilities if the Tanner graph is cycle free. Under the "cycle free assumption", the analysis of the decoding algorithm is straightforward because the incoming messages to every node are independent. In sum-product decoding the extrinsic message from a check node to a variable node, u, at the ℓth iteration, is

$$u^{(\ell)} = 2 \cdot \tanh^{-1}\left(\prod_{j=1}^{d_{c-1}} \tanh \frac{v_j^{(\ell)}}{2}\right), \tag{2}$$

where d_c is the check node degree and $v_j, j = 1, \cdots, d_c - 1$ are the received messages from all neighbors of the check node except the variable node that gets the message u. The message from a variable node to a check node, v at the ℓth iteration is equal to

$$v^{(\ell)} = \sum_{i=1}^{d_v - 1} u_i^{(\ell-1)} + u_0, \tag{3}$$

where d_v is the variable node degree, $u_i, i = 1, \cdots, d_v - 1$ are the received messages from all neighbors of the variable node except the check node that gets the message v with $u_i^{(0)} = 0$ and u_0 is the input a priori LLR of the output bit associated with the variable node. For an AWGN channel the a priori LLRs are given by $u_0 = \frac{2}{\sigma^2} y$. The total LLR of the i-th bit is $\text{LLR}_i^{(\ell)} = \sum_{i=1}^{d_v} u_i^{(\ell)} + u_0$. The i-th bit is decided to be a 0 if $\text{LLR}_i > 0$, and 1 otherwise. The decoding process ends when the decoded sequence is a codeword or until the maximum number of iterations is reached.

Density evolution is an algorithm where the evolution of probability density functions of the exchanged messages are tracked through the message-passing algorithm. It determines the behavior of an ensemble of Tanner graphs if the channel is memoryless and under the assumption that the Tanner graphs are all cycle free. Due to the symmetry of the channel and the decoder, the density evolution equations can be derived without loss of generality by assuming that all-zero codeword is sent through the channel $\big(x_i = +1, \forall i \in \{1, \cdots, n\}\big)$. Thus, negative messages indicates errors. In particular, the evolution of the error probability can be determined, via density evolution, as a function of the iteration number of the message-passing decoding algorithm. The density evolution algorithm enables to compute the *noise threshold* of a family of LDPC codes which is the maximum level (e.g. variance) of channel noise such that the probability of error converges to zero as the number of iterations tends to infinity.

Chung *et al.* investigated in [3] the sum-product decoding of LDPC codes using a Gaussian (for regular LDPC codes), or a Gaussian mixtures (for irregular LDPC codes), approximation for message densities (of u and v) under density evolution to simplify the analysis of the decoding algorithm and the design of irregular LDPC codes for AWGN channels. The authors show that the mean of a Gaussian density, which is a one-dimensional quantity, can act as faithful surrogate for the message density, which is an infinite-dimensional vector.

In order to present some results of [3] on the density evolution using Gaussian approximation, assume that the all-zero codeword is sent through the channel. Thus, the LLR message $u_0 = \frac{2}{\sigma^2}y$ from the channel is Gaussian with mean $\frac{2}{\sigma^2}$ and variance $\frac{4}{\sigma^2}$. The *symmetry condition* for a Gaussian variable with mean m and variance σ^2 reduces to $\sigma^2 = 2m$, thus we need only to keep the mean during the density evolution process. We denote the means of the messages u and v by m_u and m_v respectively. The Gaussian approximation method in [3] for irregular LDPC codes assumes that the individual output of a variable or a check node is Gaussian. Thus, the mean of the output of a variable node of degree i at the ℓth iteration, $m_{v,i}^{(\ell)}$, is given by

$$m_{v,i}^{(\ell)} = m_{u_0} + (i-1)m_u^{(\ell-1)}, \tag{4}$$

where m_{u_0} and $m_u^{(\ell-1)}$ are the means of u_0 and $u^{(\ell-1)}$ respectively. Therefore, a message v sent by the variable node to its neighbors check nodes at the ℓth iteration has a density function $f_v^{(\ell)}$ following a Gaussian mixture:

$$f_v^{(\ell)} = \sum_{i=2}^{d_v} \lambda_i \mathcal{N}(m_{v,i}^{(\ell)}, 2m_{v,i}^{(\ell)}) \tag{5}$$

Using (2), the authors demonstrate also in [3] that the mean of the Gaussian output message $u_j^{(\ell)}$ of a degree-j check node at the ℓth iteration, $m_{u,j}^{(\ell)}$ can be written as:

$$m_{u,j}^{(\ell)} = \phi^{-1}\left(1 - \left[1 - \sum_{i=2}^{d_v} \lambda_i \phi(m_{v,i}^{(\ell)})\right]^{j-1}\right), \tag{6}$$

where

$$\phi(x) = \begin{cases} 1 - \frac{1}{\sqrt{4\pi x}} \int_{\mathbb{R}} \tanh \frac{u}{2} e^{-\frac{(u-x)^2}{4x}} du & \text{if } x > 0 \\ 1, & \text{if } x = 0. \end{cases} \tag{7}$$

Hence, the mean of $u^{(\ell)}$ at the ℓth iteration, $m_u^{(\ell)}$, is obtained by linearly combining $m_{u,j}^{(\ell)}$ with weights $\rho_j, 2 \leq j \leq d_c$:

$$m_u^{(\ell)} = \sum_{j=2}^{d_c} \rho_j \phi^{-1}\left(1 - \left[1 - \sum_{i=2}^{d_v} \lambda_i \phi(m_{v,i}^{(\ell)})\right]^{j-1}\right). \tag{8}$$

The noise threshold σ^* is the supremum of all $\sigma \in \mathbb{R}^+$ such that $m_u^{(\ell)} \to \infty$ as $\ell \to \infty$. In [3], the functions $h_i(s,r)$ and $h(s,r)$ are defined as:

$$h_i(s,r) = \phi\left(s + (i-1)\sum_{j=2}^{d_c} \rho_j \phi^{-1}(1-(1-r)^{j-1})\right), \tag{9}$$

$$h(s,r) = \sum_{i=2}^{d_v} \lambda_i h_i(s,r). \tag{10}$$

The Eq. (8) is written equivalently in [3] as

$$r_\ell = h(s, r_{\ell-1}), \tag{11}$$

where $s = m_{u_0} = \frac{2}{\sigma^2}$, and $r_0 = \phi(s)$. It is demonstrated in [3] that the convergence condition is equivalent to $r_\ell \underset{\ell\to\infty}{\to} 0$ and is satisfied iff $r > h(s, r), \forall r \in (0, \phi(s))$.

2.1 Unequal Power Allocation

Problem Formulation. Given an irregular LDPC code with a variable node degree distribution $\Lambda(x) = \sum_{i=2}^{d_v} \Lambda_i x^i$, we denote by P_i the power allocated at the transmitter to a symbol associated to a variable node of degree i. Thus $P_i > 0$ if $\Lambda_i > 0$ (Λ_i is the portion of variable nodes of degree i). The bits 0 and 1 are mapped into $+\sqrt{P_i}$ and $-\sqrt{P_i}$ respectively. We assume that the destination is aware of the power allocation strategy at the source. Without loss of generality, we assume a total power constraint $P = 1$. The power constraint at the transmitter can be written as $\sum_{i=2}^{d_v} \Lambda_i P_i = 1$. In this work, we propose to choose $P_i, i \in \{2, \cdots, d_v\}$ in order to optimize the threshold of the irregular LDPC code family under consideration via density evolution. This is because density evolution, using Gaussian approximation, is a simple tool to evaluate the asymptotic performance of a family of LDPC codes. For convenience, we denote by $\mathbf{P}$ the vector whose elements are $P_i, i \in \{2, \cdots, d_v\}$. Thus, the optimization problem under consideration is the following:

$$\begin{aligned} \sigma_{th} &= \max_{\mathbf{P}} \sigma^*(\mathbf{P}) \\ \text{subject to} \quad & \sum_{i=2}^{d_v} \Lambda_i P_i = 1, \end{aligned} \tag{12}$$

where $\sigma^*(\mathbf{P})$ is the noise threshold for a given $\mathbf{P}$. When a 0-bit is transmitted with a power P_i, the mean of the message from the channel, $m_{u_{0i}} = \frac{2}{\sigma^2} P_i$ is varying with the node degree unlike the case with equal power allocation. We can easily extend the equations of the evolution of message means using Gaussian approximation in (4–11) to the case with unequal power allocation at the transmitter. Therefore, we can demonstrate that these equations become:

$$m_{v,i}^{(\ell)} = m_{u_{0i}} + (i-1) m_u^{(\ell-1)}, \tag{13}$$

$$m_v^{(\ell)} = \sum_{i=2}^{d_v} \lambda_i m_{v,i}^{(\ell)}, \tag{14}$$

$$m_{u,j}^{(\ell)} = \phi^{-1}\left(1 - \left[1 - \sum_{i=2}^{d_v} \lambda_i \phi(m_{v,i}^{(\ell)})\right]^{j-1}\right), \tag{15}$$

$$m_u^{(\ell)} = \sum_{j=2}^{d_c} \rho_j \phi^{-1}\left(1 - \left[1 - \sum_{i=2}^{d_v} \lambda_i \phi(m_{v,i}^{(\ell)})\right]^{j-1}\right). \tag{16}$$

$$h_i(s_i, r) = \phi\left(s_i + (i-1)\sum_{j=2}^{d_c} \rho_j \phi^{-1}(1 - (1-r)^{j-1})\right), \tag{17}$$

$$h(\mathbf{s}, r) = \sum_{i=2}^{d_v} \lambda_i h_i(s_i, r). \tag{18}$$

$$r_\ell = h(\mathbf{s}, r_{\ell-1}), \tag{19}$$

where $s_i = m_{u_{0i}} = \frac{2}{\sigma^2} P_i$, $\mathbf{s} = \{s_i\}$, and $r_0 = \sum_{i=2}^{d_v} \lambda_i \phi(s_i)$. The convergence condition to the threshold is the same of [3]. Therefore, for a given vector $\mathbf{P}$, the threshold $\sigma^*(\mathbf{P})$ is the supremum of all $\sigma \in \mathbb{R}^+$ such that $r_\ell \to 0$ as $\ell \to \infty$.

Proposed Solution. In order to solve the optimization problem in (12), we propose an algorithm with less complexity comparing to exhaustive search, in which the number of optimization variables is independent of the LDPC code. The proposed solution is inspired from the expression of r_ℓ in (19). Indeed, we rewrite r_ℓ as

$$r_\ell(\mathbf{P}, \sigma) = \sum_{i=2}^{d_v} \lambda_i \phi\left(\frac{2}{\sigma^2} P_i + (i-1) k_{\ell-1}\right), \tag{20}$$

where $k_{\ell-1} = \sum_{j=2}^{d_c} \rho_j \phi^{-1}(1 - (1 - r_{\ell-1})^{j-1}) \in (0, r_0)$. We recall that $\phi(x)$ is continuous and monotonically decreasing on $[0, +\infty)$, with $\phi(0) = 1$ and $\phi(\infty) = 0$ [3]. Hence $r_\ell \geq 0$. Since the convergence condition to the threshold requires that $r_\ell \underset{\ell \to \infty}{\longrightarrow} 0$, we consider the parametric family of functions $\{f_k, k \geq 0\}$ with parameter k, where f_k is defined by

$$f_k(\sigma, \mathbf{P}) = \sum_{i=2}^{d_v} \lambda_i \phi\left(\frac{2}{\sigma^2} P_i + (i-1) k\right), \tag{21}$$

and for a fixed σ, we look only for the vectors $\mathbf{P}$ which are the minimas of $\{f_k\}_{k \geq 0}$. Therefore, for a fixed σ and k, we consider the following optimization problem

$$\begin{aligned} \mathbf{P}^*(\sigma, k) &= \arg\min_{\mathbf{P}} f_k(\sigma, \mathbf{P}) \\ \text{subject to} \quad & \sum_{i=2}^{d_v} \Lambda_i P_i = 1. \end{aligned} \tag{22}$$

Then, we define $r(k, \sigma) = r_\infty(\mathbf{P}^*(\sigma, k), \sigma)$ which can be calculated using (20). For a fixed k, the threshold $\sigma^*(k)$ is defined as the maximal value of σ such that $r(k, \sigma) \to 0$. Finally, k is chosen to maximize the threshold, thus we denote

$$k^* = \arg\max_k \sigma^*(k). \tag{23}$$

For a given k and σ, we should solve the optimization problem in (22). The expression of $\phi(x)$ in (7) makes very difficult to have a closed form expression of $\mathbf{P}^*(\sigma, k)$. In [3] the following approximation of $\phi(x)$ is used $\phi(x) \sim e^{-\alpha x^\gamma + \beta}$, where $\alpha = 0.4527$, $\beta = 0.0218$ and $\gamma = 0.86$. Even with this approximation, it is difficult to obtain an analytic solution for the optimization problem (22). Therefore, we propose to approximate $\phi(x)$ by a convex function of the form $\phi(x) \sim e^{-ax}$ with $a > 0$. Since the value for which the function ϕ should be evaluated in (21) depends on i and since e^{-ax} cannot approximate exactly $e^{-\alpha x^\gamma + \beta}$ for all values of x, a, we define a function $\phi_i(x) = e^{-a_i x}$ for each $i \in \{2, \cdots, d_v\}$ with $a_i > 0$. Thus, using the latest approximation, the optimization algorithm in (22) becomes:

$$\mathbf{P}^*(\sigma, k) = \arg\min_{\mathbf{P}} \sum_{i=2}^{d_v} \lambda_i \phi_i\left(\frac{2}{\sigma^2} P_i + (i-1)k\right)$$
$$\text{subject to} \quad \sum_{i=2}^{d_v} \Lambda_i P_i = 1. \tag{24}$$

Proposition 1. *The solution of the optimization algorithm in (24) is*

$$P_i^*(\sigma, k) = -\frac{(i-1)k}{k_0} - \frac{1}{a_i k_0} \log\left(\frac{\Lambda_i}{a_i \lambda_i k_0}\right)$$
$$- \left(\frac{1}{a_i \sum_i \frac{\Lambda_i}{a_i}}\right)\left(-1 - k \sum_{i=2}^{d_v} (i-1)\frac{\Lambda_i}{k_0} - \sum_{i=2}^{d_v} \frac{\Lambda_i}{a_i k_0} \log\left(\frac{\Lambda_i}{a_i \lambda_i k_0}\right)\right), \tag{25}$$

where $k_0 \triangleq \frac{2}{\sigma^2}$ *and* $i \in \{2, \cdots, d_v\}$.

Proof. First, we form the Lagrangian of problem (24)

$$L(\mathbf{P}, \theta; \sigma, k) = \sum_{i=2}^{d_v} \lambda_i \phi_i\left(\frac{2}{\sigma^2} P_i + (i-1)k\right) + \theta\left(\sum_{i=2}^{d_v} \Lambda_i P_i - 1\right). \tag{26}$$

where θ is the Lagrange multiplier. Since the objective function in problem (24) is convex and the constraint is affine, the KKT conditions guarantee the global optimality of the solution. Thus, the solution $(\mathbf{P}^*, \theta^*)$ of problem (24) verifies the following equations

$$\begin{cases} \frac{\partial L}{\partial P_i}(P_i^*) = -a_i k_0 \lambda_i e^{-a_i(k_0 P_i^* + (i-1)k)} + \theta^* \Lambda_i = 0, & \forall i \in \{2, \cdots, d_v\} \\ \sum_{i=2}^{d_v} \Lambda_i P_i^* - 1 = 0. \end{cases} \tag{27}$$

For simplicity of notation, we drop the dependence of P_i^* on σ and k. From the first equation in (27), we get

$$P_i^* = -\frac{k(i-1)}{k_0} - \frac{\log\left(\frac{\theta^* \Lambda_i}{a_i \lambda_i k_0}\right)}{a_i k_0}. \tag{28}$$

After substituting (28) in the equation $\sum_{i=2}^{d_v} \Lambda_i P_i^* - 1 = 0$, we get

$$\theta^* = e^{\frac{-k\sum_i (i-1)\Lambda_i - k_0 - \sum_i \frac{\Lambda_i}{a_i}\log\left(\frac{\Lambda_i}{a_i \lambda_i k_0}\right)}{\sum_i \frac{\Lambda_i}{a_i}}}. \quad (29)$$

Finally, after substituting (29) in (28), we get (25).

In order to determine the set $\{a_i\}_{i>1}$, we set $e^{-a_i x} = e^{-\alpha x^\gamma + \beta}$ where x is the value for which the function ϕ should be evaluated. Thus $a_i = \frac{-\alpha x^\gamma + \beta}{-x}$. Since we should evaluate $\phi_i(x)$ in (24) where $x = \frac{2}{\sigma^2}P_i + (i-1)k$ and that P_i is unknown, we set in our experiments $a_i = \frac{-\alpha \hat{x}^\gamma + \beta}{-\hat{x}}$ where $\hat{x} = \frac{2}{\sigma^2} + (i-1)k$.

The proposed solution is summarized in Algorithm 1.

Remark: In the optimization problem (24), we discarded the constraint $P_i \geq 0$. The solution obtained in Proposition 1 can be written as $P_i^* = A_i + \frac{1}{k_0}B_i$ where $A_i > 0$ and $k_0 = 2/\sigma^2$. Thus if for some $i \in \{2, \cdots, d_v\}$ and k, the obtained solution P_i^* is non-positive, we should decrease the value of σ while searching for the threshold, as stated in Algorithm 1.

Algorithm 1. Algorithm to solve (12)

```
1: $\epsilon \leftarrow 10^{-10}$, MaxIter $\leftarrow 10^4$.
2: for $k = 0 : k_{\max}$ do
3:     $\sigma_{\min} \leftarrow 0$, $\sigma_{\max} \leftarrow 10$
4:     $\sigma \leftarrow \frac{\sigma_{\max}+\sigma_{\min}}{2}$
5:     while $\sigma_{\max} - \sigma_{\min} > \epsilon$ do
6:         Compute $\mathbf{P}^*(\sigma, k)$ using Proposition 1
7:         if there exists at least one $P_i^* < 0$ then decrease $\sigma_{\max}$
8:         else
9:             $m_{u_{0i}} \leftarrow \frac{2}{\sigma^2}P_i^*(\sigma, k)$, for $i \in \{2, \cdots, d_v\}$
10:            for $\ell = 1$ : MaxIter do
11:                Calculate $r_\ell = h(\mathbf{s}, r_{\ell-1})$ using (19)
12:                if $r_\ell < \epsilon$ then break
13:            if $r_\ell < \epsilon$ then $\sigma_{\min} \leftarrow \sigma$
14:            else $\sigma_{\max} \leftarrow \sigma$
15:            $\sigma \leftarrow \frac{\sigma_{\max}+\sigma_{\min}}{2}$
16:    Save $\sigma^*(k) = \sigma$
17: $\sigma_{\text{th}} \leftarrow \max_k \sigma^*(k)$
```

In Algorithm 1, "MaxIter" represents the maximum number of iterations that can be performed by the decoder. In our work, we will study also the unequal power allocation for decoders with limited number of iterations L. In this case, we set MaxIter= L in Algorithm 1. Finally, we should note that the power allocation vector maximizing the noise threshold in Algorithm 1 to get zero error probability could achieve worst performance than equal power allocation for practical (low) bit error rate values (BER). Hence we propose to optimize the noise threshold

$\sigma_{\text{th}}(\eta)$ for a target error probability η. Using the density evolution algorithm with Gaussian approximation, the error probability at the Lth iteration of the decoder, $P_e^{(L)}$, can be calculated. Therefore, for a target BER η, the convergence condition to the threshold $\sigma_{\text{th}}(\eta)$ becomes $P_e^{(L)} < \eta$, instead of $r_L \to 0$.

3 Unequal Power Allocation for Relay Channels

In this section, we consider the Gaussian relay channel case. The indexes s, r and d will refer to the source, the relay and the destination respectively. The relay uses the "amplify and forward" strategy. Without loss of generality, we consider an average power constraint at the source $P_s = 1$ and an average power constraint at the relay $P_r = 1$.

In the relay channel case, the source and the relay can use simultaneously different power allocation strategies but this makes the problem difficult to solve due to the large number of variables. Therefore, we will study in the following, a strategy where the source only uses an unequal power allocation strategy. The transmitted signal x_s by the source is given by $x_s = \sqrt{P_{si}}x$ where $x \in \{+1, -1\}$ and P_{si} is the power allocated at the source for a bit associated to variable node of degree i. We denote by $\mathbf{P_s}$ the vector whose elements are P_{si}, $i \in \{2, \cdots, d_v\}$. The received signal by the relay is $y_{sr} = \sqrt{P_{si}}x + z_{sr}$, $z_{sr} \sim \mathcal{N}(0, \sigma_{sr}^2)$. The transmitted signal by the relay is given by $x_r = f y_{sr}$, where $f = \frac{1}{\sqrt{P_s + \sigma_{sr}^2}} = \frac{1}{\sqrt{1+\sigma_{sr}^2}}$. The destination receives $y_{sd} = x_s + z_{sd}$ from the source ($z_{sd} \sim \mathcal{N}(0, \sigma_{sd}^2)$) and $y_{rd} = x_r + z_{rd} = f\sqrt{P_{si}}x + f z_{sr} + z_{rd}$ from the relay ($z_{rd} \sim \mathcal{N}(0, \sigma_{rd}^2)$). The destination, aware of the power allocation strategy, can determine y'_{sd} and y'_{rd} as $y'_{sd} = \frac{y_{sd}}{\sqrt{P_{si}}} = x + z'_{sd}$, where $z'_{sd} = \frac{z_{sd}}{\sqrt{P_{si}}} \sim \mathcal{N}(0, \frac{\sigma_{sd}^2}{P_{si}})$, and $y'_{rd} = \frac{y_{rd}}{f\sqrt{P_{si}}} = x + z'_{rd}$, where $z'_{rd} = \left(\frac{z_{sr}}{\sqrt{P_{si}}} + \frac{z_{rd}}{f\sqrt{P_{si}}}\right) \sim \mathcal{N}(0, \frac{\sigma_{sr}^2}{P_{si}} + \frac{\sigma_{rd}^2}{f^2 P_{si}})$. At the destination, the LLR message at a variable node of degree i from the relay channel is given by

$$u_{0i} = \log \frac{p(x = +1|y'_{sd}, y'_{rd})}{p(x = -1|y'_{sd}, y'_{rd})} \tag{30}$$

$$= \underbrace{\log \frac{p(y'_{sd}|x = +1)}{p(y'_{sd}|x = -1)}}_{} + \underbrace{\log \frac{p(y'_{rd}|x = +1)}{p(y'_{rd}|x = -1)}}_{} \tag{31}$$

$$= \quad u_{0i}^s \quad + \quad u_{0i}^r \tag{32}$$

where $u_{0i}^s = \frac{2y'_{sd}P_{si}}{\sigma_{sd}^2}$ and $u_{0i}^r = \frac{2y'_{rd}P_{si}}{\sigma_{sr}^2 + \frac{\sigma_{rd}^2}{f^2}}$. Hence, when the all-zero codeword is assumed to be sent by the source, the mean of the LLR message u_{0i} is given by $m_{u_{0i}} = m_{u_{0i}^s} + m_{u_{0i}^r}$, where $m_{u_{0i}^s} = \frac{2P_{si}}{\sigma_{sd}^2}$ and $m_{u_{0i}^r} = \frac{2P_{si}}{\sigma_{sr}^2 + \frac{\sigma_{rd}^2}{f^2}}$. In the relay channel case, we have three independent channels. Therefore, we should fix the SNR of two channels and find the power allocation vector which maximizes the noise threshold for the remaining channel. In this work, we consider the setting where the source and the relay know both the SNRs of the channels source-relay and source-destination. The power allocation vector at the source is optimized in order to maximize the threshold of the relay-destination channel. This optimization can be solved in the same manner as in the point-to-point case. Indeed, the Algorithm 1 can be extended for the relay channel case after replacing the expression of $m_{u_{0i}}$ in Algorithm 1 by its expression in the relay case depending on the strategy used. The output σ_{th} of Algorithm 1 will refer to $\sigma_{rd_{\text{th}}}$ in the relay case. Moreover, the Proposition 1 should be updated for the relay channel case in Algorithm 1. Hence, it is easy to demonstrate that the expression of P^*_{si} as function of k and the channel noise variances is obtained from Proposition 1 by replacing k_0 with $k_0 = \frac{2}{\sigma^2_{sd}} + \frac{2}{\sigma^2_{sr} + \frac{\sigma^2_{rd}}{f^2}}$.

4 Simulation Results and Discussion

This section presents simulation results on unequal power allocation for irregular LDPC codes. Table 1 shows the SNR threshold of some irregular LDPC codes using Gaussian approximation with both equal power allocation and unequal power allocation. It gives also the power allocation function $P(x)$ obtained by Algorithm 1, for each code in Table 1, where $P(x) = \sum_i P_i x^{i-1}$ and P_i is the power allocated for the degree-i variable node. We observed in our simulations that our proposed algorithm gives the same threshold values as the exhaustive search method. We observe in Table 1 that a gain up to 0.22 dB can be obtained on the threshold with respect to equal power allocation strategy. However, for some codes EPA is optimal. We recall that the proposed power allocation method in Algorithm 1 relies on computing the threshold $\text{SNR}^*(k)$ for each parameter $k \geq 0$ and then to choose k^* which gives the better threshold (cf. (22) and 23). Figure 1 shows $\text{SNR}^*(k)$ as a function of k for the LDPC code of rate 1/2 and $\Lambda(x) = 0.5x^2 + 0.5x^4$. We observe that the SNR threshold decreases with k until k^*. Intuitively, k is small means that the density evolution algorithm is far from the convergence (cf. 20). Thus, the power allocation vector optimized for small values of k could be not the optimal power allocation vector when the density evolution algorithm is near convergence. Moreover, when k becomes large, the objective function to minimize in problem (24) decreases (since $\phi(x)$ is decreasing with x), and the objective function becomes less dependent on the power allocation vector (since its value is close to zero when k is large). Thus there is a trade-off in the optimal value of k which minimizes the SNR threshold.

Remark: for k and σ fixed, the objective function to minimize in problem (24) can be written as $f(\mathbf{P}) = \sum_{i=2}^{d_v} \lambda_i f_i(P_i)$, where $f_i(P_i) = \phi_i\big(\frac{2}{\sigma^2}P_i + (i-1)k\big)$. When the irregular LDPC code involves a degree i which is too large, $f_i(P_i)$ becomes too small and the value of P_i will not affect too much the value of the objective function. Thus, in our simulations, we define a "saturation" parameter K_{sat} and we set $f_i(P_i) = \phi_i\big(\frac{2}{\sigma^2}P_i + K_{\text{sat}}\big)$, $\forall(i,k)$ such that $(i-1)k \geq K_{\text{sat}}$. This introduces an additional optimization variable in Algorithm 1 (K_{sat}), however, this value is optimized in our simulations one time for all the simulated codes.

Table 1. The threshold in dB of some LDPC code families of rate 1/2 with EPA and UPA for the BIAWGN point-to-point channel.

$\Lambda(x)$	SNR$_{th}$ EPA	SNR$_{th}$ UPA	$P(x)$
$0.5x^2 + 0.5x^4$	0.8733	0.8725	$0.9775x^2 + 1.0225x^4$
$0.5x^2 + 0.5x^6$	1.0854	0.9492	$0.8183x^2 + 1.1817x^6$
$0.5x^2 + 0.5x^8$	1.4520	1.2301	$0.8509x^2 + 1.1491x^8$

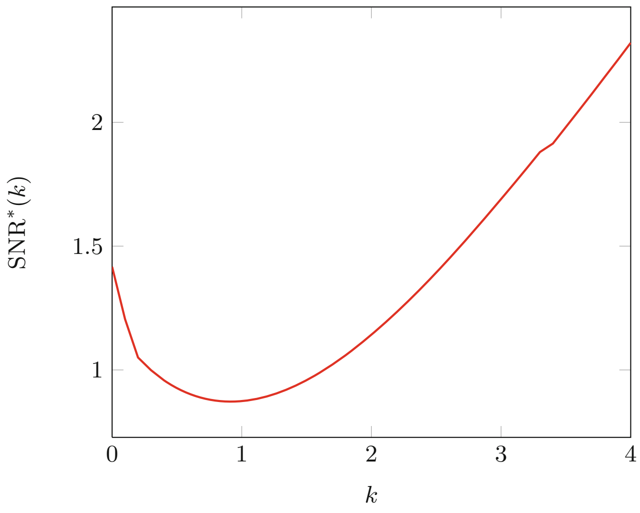

Fig. 1. SNR$^*(k)$ as a function of k, for the code of rate 1/2 and $\Lambda(x) = 0.5x^2 + 0.5x^4$.

Figure 2 shows the SNR threshold as a function of the decoder maximum number of iterations L for a target BER less than 10^{-8}. We observe that a gain of 0.5 dB on the SNR threshold can be obtained by optimizing the power allocation with respect to equal power allocation. We should note that in this work, we optimize the power allocation for a given code but rather it is also possible to optimize the code degree distributions for a decoder of fixed number of iterations L and using EPA. Figure 3 shows the SNR threshold for the relay-destination channel as a function of the decoder maximum number of iterations L for a target BER less than 10^{-8}, when $\sigma_{sd} = 1.1$ and $\sigma_{sr} = 0.7$. The gain of the unequal power allocation method with respect to the equal power allocation seems to be more important for the relay case (up to 2.4 dB). This is because we have more degrees of freedom in the relay case where we can fix the SNR of two channels and determine the threshold for the remaining channel. When L increases, the gain brought by UPA decreases in Fig. 2 and Fig. 3.

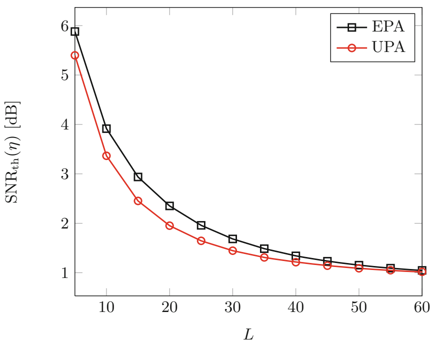

Fig. 2. $\mathrm{SNR}_{\mathrm{th}}(\eta = 10^{-8})$ as a function of L, for the code of rate 1/2 and $\Lambda(x) = 0.5x^2 + 0.5x^4$, in the point-to-point channel case.

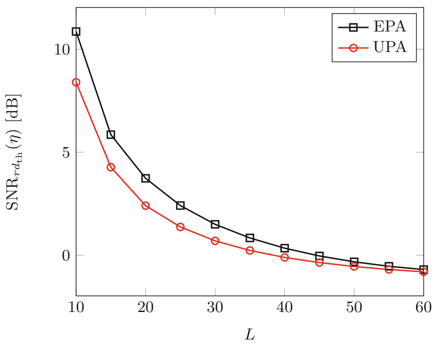

Fig. 3. $\mathrm{SNR}_{rd_{\mathrm{th}}}(\eta = 10^{-8})$ as a function of L, for the code of rate 1/2 and $\Lambda(x) = 0.5x^2 + 0.5x^4$, in the relay channel case with $\sigma_{sd} = 1.1$ and $\sigma_{sr} = 0.7$.

5 Conclusion

In this paper, we investigated an unequal power allocation method for irregular LDPC codes where the power allocated of a coded bit depends on the degree of its associated variable node in the Tanner graph. We proposed an algorithm to optimize the power allocation vector using a modified density evolution algorithm under the Gaussian approximation. Simulation results show that in some cases, the unequal power allocation leads to a gain on the threshold of the LDPC code comparing to equal power allocation.

References

1. Richardson, T.J., Shokrollahi, M.A., Urbanke, R.L.: Design of capacity-approaching irregular low-density parity-check codes. IEEE Trans. Inf. Theor. **47**(2), 619–637 (2001)
2. Qi, H., Malone, D., Subramanian, V.: Does every bit need the same power? An investigation on unequal power allocation for irregular LDPC codes. In: WCSP, Nanjing, pp. 1–5 (2009)
3. Chung, S.Y., Richardson, T., Urbanke, R.L.: Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation. IEEE Trans. Inf. Theor. **47**(2), 657–670 (2001)