Technical University of Crete

Department of Electronic and Computer Engineering

Design of LDPC Codes for the Two-User Gaussian Multiple Access Channel

By:

Alexios-Konstantinos Balatsoukas-Stimming

Submitted in June 2012 in partial fulfilment of the requirements for the Master of Science degree in Electronic and Computer Engineering

Advisor: Professor Athanasios P. Liavas

Co-advisor: Professor Nikos D. Sidiropoulos

Co-advisor: Assistant Professor George N. Karystinos

Abstract

The capacity region of Gaussian Multiple Access Channels (GMACs) has been known since 1971. The efficient design of powerful codes that can achieve points near the dominant face of the capacity region, where the sum-rate is maximal, is an interesting problem. Significant progress has been made in this direction using time sharing, rate-splitting, as well as joint iterative decoding. Joint iterative decoding seems to be the most promising path, especially for codes that have low-complexity decoders, like Low-Density Parity-Check (LDPC) codes.

LDPC codes are capacity-approaching over a wide variety of channels. Additionally, elegant tools, such as Density Evolution and EXIT charts, can be used to accurately predict the asymptotic performance of an LDPC code ensemble. These tools can be used for the design of optimal LDPC codes, allowing for transmission over many types of channels with vanishingly small probability of error.

In this thesis, we focus on the two-user GMAC. To the best of our knowledge, there exist two LDPC code design frameworks for this channel. Amraoui *et al.* use Density Evolution, which is very demanding in terms of computational complexity. Roumy and Declercq use EXIT charts, a low-complexity approximation of Density Evolution, but their design is restricted to the case where the power of both users at the receiver is equal. We extend the EXIT chart based optimization framework by removing the equal power constraint, allowing for the optimization of LDPC codes over unequal power two-user GMACs. We show that, under some assumptions, the optimization problem can be expressed as an alternating linear programming problem, which can be solved efficiently. The resulting codes are close to optimal, in terms of sum-rate, and exhibit very good finite-length behavior.

Acknowledgments

I would like to take some time here and thank the people and organizations who made the completion of this work possible.

First of all, my parents, without whom my entire studies would not have been possible. I would also like to thank my thesis supervisor, Professor Athanasios P. Liavas, for all his guidance, support and patience. He taught me to pursue a deep understanding of any subject I am studying and, being a bit of a stickler for accuracy, how to write proper technical reports. Most importantly, I learned not to quit in the face of difficulty. Many thanks also go to my thesis committee, Professor Nikos D. Sidiropoulos and Assistant Professor George N. Karystinos, who provided their expertise and advice in all matters whenever needed. Last, and exactly the opposite of "least", my wonderful girlfriend Kynthia without whom I would never have made it this far.

I would also like to warmly thank the Alexander S. Onassis Public Benefit Foundation for supporting my postgraduate studies with scholarship G ZG 028/2010-2011, and Ms. Konstantina Zarkali in particular for always being very helpful with administrative issues.

"Would you like the hotel moved a bit to the left?"
– Basil Fawlty

Contents

List of Figures

Chapter 1

Introduction

1.1 Gaussian Multiple Access Channels (GMACs)

In a two-user multiple access setting, a channel is used by two users who wish to simultaneously send information to a common receiver. Let the two length-n codewords for users 1 and 2 be denoted as $\mathbf{c}^{[1]} \in \mathcal{C}_1$ and $\mathbf{c}^{[2]} \in \mathcal{C}_2$, respectively, where $\mathcal{C}_1$ and $\mathcal{C}_2$ denote the corresponding codebooks. Since the two users transmit independent information, $\mathbf{c}^{[1]}$ and $\mathbf{c}^{[2]}$ are statistically independent. Let the BPSK modulated codewords be denoted as $\mathbf{x}^{[1]}$ and $\mathbf{x}^{[2]}$, where we employ the standard mapping

$$\mathbf{x}^{[1]} = 1 - 2\mathbf{c}^{[1]} \quad \text{and} \quad \mathbf{x}^{[2]} = 1 - 2\mathbf{c}^{[2]}. \tag{1.1}$$

Frame synchronized transmission through an equal power two-user Gaussian MAC (GMAC) can be modeled as

$$\mathbf{y} = \mathbf{x}^{[1]} + \mathbf{x}^{[2]} + \mathbf{w}, \quad \mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}), \tag{1.2}$$

where $\mathbf{y}$ denotes the observation vector at the receiver and $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ denotes the multivariate Gaussian distribution with mean vector $\boldsymbol{\mu} = \mathbf{0}$ and covariance matrix $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$. Two-user GMAC channels can exist both on their own and as parts of more complex communication channels. The most prominent example of a complex system that contains a MAC is the relay channel [1]. Although the equal power assumption is rather constraining, such channels can exist in practice if proper power control is performed at the transmitters.

We can lift the equal power assumption by assigning powers $\tilde{P}_1$ and $\tilde{P}_2$ to users 1 and 2, respectively, leading to the following channel model

$$\mathbf{y} = \sqrt{\tilde{P}_1}\mathbf{x}^{[1]} + \sqrt{\tilde{P}_2}\mathbf{x}^{[2]} + \tilde{\mathbf{w}}, \quad \tilde{\mathbf{w}} \sim \mathcal{N}(0, \sigma^2\mathbf{I}). \tag{1.3}$$

In the sequel, it will be convenient to normalize the users' powers with respect to the noise variance, i.e.

$$\mathbf{y} = \sqrt{\frac{\tilde{P}_1}{\sigma^2}}\mathbf{x}^{[1]} + \sqrt{\frac{\tilde{P}_2}{\sigma^2}}\mathbf{x}^{[2]} + \mathbf{w}, \quad \mathbf{w} \sim \mathcal{N}(0, \mathbf{I}). \tag{1.4}$$

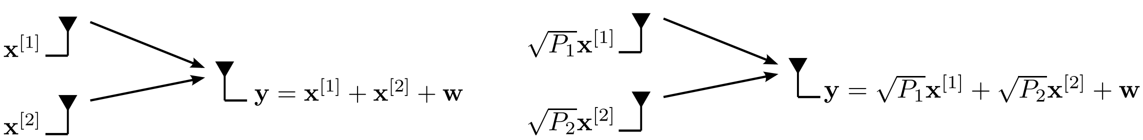


Figure 1.1: Equal and unequal power two-user GMACs.

By defining $P_1 \triangleq \frac{\tilde{P}_1}{\sigma^2}$ and $P_2 \triangleq \frac{\tilde{P}_2}{\sigma^2}$, we get the following expression

$$\mathbf{y} = \sqrt{P_1}\mathbf{x}^{[1]} + \sqrt{P_2}\mathbf{x}^{[2]} + \mathbf{w}, \quad \mathbf{w} \sim \mathcal{N}(0, \mathbf{I}). \tag{1.5}$$

Using the above model, we define the SNR for each user as

$$\mathrm{SNR}_1 \triangleq P_1, \quad \text{and} \quad \mathrm{SNR}_2 \triangleq P_2. \tag{1.6}$$

1.2 Low-Density Parity-Check (LDPC) Codes

Low-Density Parity-Check (LDPC) codes are a class of capacity approaching linear block codes. They were invented by Robert Gallager in 1960 [2] but forgotten for over 30 years, mainly due to practical issues regarding their implementation. Since their revival in 1995 by David MacKay and Radford Neal [3], a very lively research community has been engaged with the study of LDPC codes. As their name implies, they have a parity-check matrix with a low density of non-zero entries, i.e. a sparse matrix. The definition of "low density" is a bit loose but, generally, matrices with less than 10% non-zero entries are considered to be sparse. The number of non-zero entries in a sparse matrix increases linearly with respect to matrix dimensions, and not quadratically as in regular (dense) matrices. It is exactly this property that enables LDPC codes to be decodable with *linear* time complexity. Under some conditions, *linear* time encoding is also possible [4].

LDPC codes, just like any linear code, can be defined through their $m \times n$ *parity-check matrix*, commonly denoted as $\mathbf{H}$. The codebook $\mathcal{C}$ consists of all vectors $\mathbf{c} \in \{0,1\}^n$, for which $\mathbf{Hc} = \mathbf{0}$, where additions are performed modulo 2. More formally

$$\mathcal{C} = \{\mathbf{c} \in \{0,1\}^n : \mathbf{Hc} = \mathbf{0}\},$$

which means that $\mathcal{C}$ is the (modulo 2) nullspace of the matrix $\mathbf{H}$.

Tanner graphs [5] provide a way to visualize the parity-check constraints imposed by an LDPC code, as well as the effect of the transmission channel and, in our case, the effect of the interfering user. Single user Tanner graphs consist of variable nodes and check nodes. A variable node j is connected to a check node i iff $H_{ij} = 1$. In words, a variable node j is connected to a check node i iff the codeword bit i participates in the parity-check equation j. Assume that the number of variable nodes of degree i is Λ_i, so that $\sum_i \Lambda_i = n$. In the same fashion, the number of check nodes of degree i is P_i, so that $\sum_i P_i = n(1-r) = m$, where r is the *design rate* of the code. The following compact notation is more convenient

$$\Lambda(x) \triangleq \sum_i \Lambda_i x^i, \qquad P(x) \triangleq \sum_i P_i x^i.$$

From the above definitions, we get the following useful relationships

$$\Lambda(1) = n, \quad P(1) = n(1-r), \quad r(\Lambda, P) = 1 - \frac{P(1)}{\Lambda(1)}, \quad \Lambda'(1) = P'(1).$$

The first four relationships are obvious. If we rewrite the left part of the last one as $\Lambda'(1) = \sum_i i\Lambda_i$, we see that this sum represents the total number of connections on the variable node side. This number is equal to the number of connections on the check node side. $\Lambda(x)$ and $P(x)$

are called the *check and variable degree distributions* from a *node perspective*. It is usually more convenient to use the corresponding normalized distributions

$$L(x) \triangleq \frac{\Lambda(x)}{\Lambda(1)} = \sum_i L_i x^i, \quad R(x) \triangleq \frac{P(x)}{P(1)} = \sum_i R_i x^i.$$

Naturally, it holds that

$$\sum_i L_i = \sum_i R_i = 1.$$

We can also define the distributions from an *edge perspective*, meaning that the polynomial coefficients represent the fraction of edges connected to degree i variable or check nodes. This definition is particularly useful for Density Evolution. The normalized node degree distributions from an edge perspective are defined as

$$\lambda(x) \triangleq \frac{L'(x)}{L'(1)} = \sum_i \lambda_i x^{i-1}, \quad \rho(x) \triangleq \frac{R'(x)}{R'(1)} = \sum_i \rho_i x^{i-1}.$$

Again, it holds that

$$\sum_i \lambda_i = \sum_i \rho_i = 1.$$

The design rate of an edge perspective degree distribution pair is defined as

$$r(\lambda(x), \rho(x)) \triangleq 1 - \frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx} = 1 - \frac{\sum_i \rho_i/i}{\sum_i \lambda_i/i}.$$

We refer the interested reader to Richardson and Urbanke [6] for further details.

1.3 Previous Work and Motivation

The capacity region of a two-user GMAC is the pentagon defined by [7]

$$\begin{aligned} R^{[1]} &\leq I(X_1;Y|X_2) = C\left(\frac{P_1}{N}\right), \\ R^{[2]} &\leq I(X_2;Y|X_1) = C\left(\frac{P_2}{N}\right), \\ R^{[1]} + R^{[2]} &\leq I(X_1,X_2;Y) = C\left(\frac{P_1+P_2}{N}\right), \end{aligned}$$

where $C(x) = \frac{1}{2}\log_2(1+x)$, P_i is the power of user i, $i = 1, 2$, and N is the noise power. In the equal power case, we have $I(X_1;Y|X_2) = I(X_2;Y|X_1)$, so the capacity region looks like the example in Figure 1.2. In the case where the users have different powers, the region is stretched in the direction of the user with the higher power.

We are mainly interested in rate pairs lying on the diagonal line, since this line represents the rate pairs with maximal sum rate. This line is also called the *dominant face*, denoted $\mathcal{D}$. The corner points of the capacity region can be easily achieved by successive decoding, i.e. by decoding one user while treating the other user as interference and then subtracting the decoded codeword from the received signal and decoding the second user [8, 9]. The remaining points of the line can be achieved either by time-sharing [7], rate-splitting [10], or by joint decoding [11, 12]. Time sharing is not implementation friendly, since, for the two user case, two codes

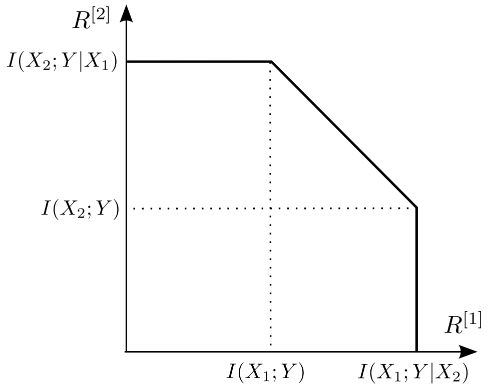


Figure 1.2: GMAC capacity region.

are needed for each user. In the general case with M users, the required number of codes is of order M^2. The discussion about rate-splitting in [10] is purely theoretical and no code design method is provided.

On the other hand, joint decoding of LDPC codes is practically feasible due to the low-complexity and excellent performance of the most widely used message-passing algorithm for the approximation of the bit-wise Maximum A Posteriori (MAP) decoding rule, i.e. Belief Propagation (BP) [13]. LDPC code design methods for the two-user GMAC using joint decoding have been proposed in [11] and [12]. However, in [11], Density Evolution, which has very high computational complexity, is used and, in [12], the analysis is restricted to the equal power GMAC.

In this thesis, we are concerned with extending the existing methodology to include the unequal power case, thus developing a more general framework for low-complexity design of LDPC codes for arbitrary power two-user GMACs.

Chapter 2

Asymptotic Analysis

2.1 Bit-Wise MAP Decoding and Belief Propagation

In this section, we derive the bit-wise MAP decoding rules for both users of the GMAC and we describe the corresponding Tanner graph. We also prove that the two-user Tanner graph is asymptotically cycle-free, like in the single user case. Furthermore, we review the variable and check node update rules for the BP algorithm and we derive the update rule for a new type of node, which can not be found in single user graphs, i.e. the *state node*.

2.1.1 Bit-Wise MAP Decoding Rule

We present a detailed derivation of the bit-wise MAP decoding rule for user 1. The MAP rule for user 2 can be derived similarly. Let $\hat{x}_i^{[1]}$ denote the MAP estimate for symbol i of user 1's BPSK modulated codeword based on the observation vector $\mathbf{y}$. This is equivalent to the MAP estimate of codeword bit i of user 1's binary codeword, since the mapping describing BPSK modulation is a bijection. By definition of the MAP rule, we have

$$\hat{x}_i^{[1]} \triangleq \arg\max_{x_i^{[1]}} p(x_i^{[1]}|\mathbf{y}), \tag{2.1}$$

where the maximum is taken over $x_i^{[1]} \in \{\pm 1\}$. Equation (2.1) can be rewritten as follows

$$\begin{aligned}
\hat{x}_i^{[1]} &\triangleq \arg\max_{x_i^{[1]}} p(x_i^{[1]}|\mathbf{y}) && (2.2)\\
&= \arg\max_{x_i^{[1]}} \sum_{\sim x_i^{[1]}} p(\mathbf{x}^{[1]}|\mathbf{y})\mathbf{1}_{[\mathbf{c}^{[1]}\in\mathcal{C}_1]} && (2.3)\\
&= \arg\max_{x_i^{[1]}} \sum_{\sim x_i^{[1]}} \sum_{\mathbf{x}^{[2]}} p(\mathbf{x}^{[1]},\mathbf{x}^{[2]}|\mathbf{y})\mathbf{1}_{[\mathbf{c}^{[1]}\in\mathcal{C}_1]}\mathbf{1}_{[\mathbf{c}^{[2]}\in\mathcal{C}_2]} && (2.4)\\
&= \arg\max_{x_i^{[1]}} \sum_{\sim x_i^{[1]}} \sum_{\mathbf{x}^{[2]}} p(\mathbf{y}|\mathbf{x}^{[1]},\mathbf{x}^{[2]})p(\mathbf{x}^{[1]})p(\mathbf{x}^{[2]})\mathbf{1}_{[\mathbf{c}^{[1]}\in\mathcal{C}_1]}\mathbf{1}_{[\mathbf{c}^{[2]}\in\mathcal{C}_2]} && (2.5)\\
&= \arg\max_{x_i^{[1]}} \frac{1}{|\mathcal{C}_1|\cdot|\mathcal{C}_2|} \sum_{\sim x_i^{[1]}} \sum_{\mathbf{x}^{[2]}} p(\mathbf{y}|\mathbf{x}^{[1]},\mathbf{x}^{[2]})\mathbf{1}_{[\mathbf{c}^{[1]}\in\mathcal{C}_1]}\mathbf{1}_{[\mathbf{c}^{[2]}\in\mathcal{C}_2]}, && (2.6)\\
&= \arg\max_{x_i^{[1]}} \sum_{\sim x_i^{[1]}} \sum_{\mathbf{x}^{[2]}} p(\mathbf{y}|\mathbf{x}^{[1]},\mathbf{x}^{[2]})\mathbf{1}_{[\mathbf{c}^{[1]}\in\mathcal{C}_1]}\mathbf{1}_{[\mathbf{c}^{[2]}\in\mathcal{C}_2]}, && (2.7)
\end{aligned}$$

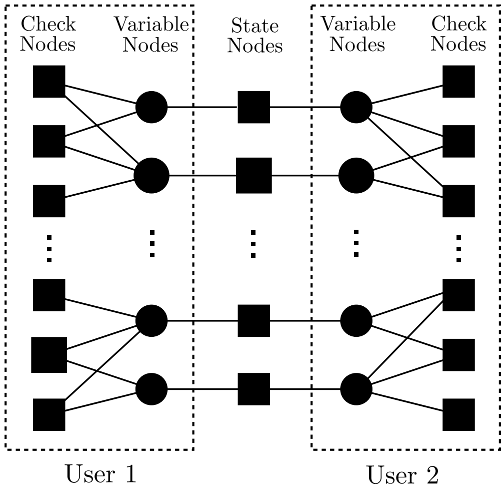


Figure 2.1: Two-user MAC Tanner Graph.

where $\sum_{\sim x_i^{[1]}}$ denotes summation over all symbols except $x_i^{[1]}$ and $|\mathcal{C}|$ denotes the cardinality of a set $\mathcal{C}$. From (2.4) to (2.5) we applied Bayes' rule and from (2.5) to (2.6) we used the fact that codewords are drawn uniformly at random from their respective codebooks, so that $\mathbf{x}^{[1]}$ has probability $\frac{1}{|\mathcal{C}_1|}$ if a $\mathbf{c}^{[1]}$ such that $\mathbf{x}^{[1]} = 1 - 2\mathbf{c}^{[1]}$ exists in $\mathcal{C}_1$, otherwise it has probability zero. The same holds for $\mathbf{x}^{[2]}$. The function $\mathbf{1}_{[\mathbf{c}\in\mathcal{C}]}$ is an indicator function defined as

$$\mathbf{1}_{[\mathbf{c}\in\mathcal{C}]} = \begin{cases} 1, & \mathbf{c} \in \mathcal{C}, \\ 0, & \text{otherwise}. \end{cases} \tag{2.8}$$

By further assuming that the channel is memoryless, we can arrive at the final expression

$$\hat{x}_i^{[1]} = \arg\max_{x_i^{[1]}} \sum_{\sim x_i^{[1]}} \sum_{\mathbf{x}^{[2]}} \left(\prod_{k=1}^{n} p(y_k | x_k^{[1]}, x_k^{[2]}) \right) \mathbf{1}_{[\mathbf{c}^{[1]}\in\mathcal{C}_1]} \mathbf{1}_{[\mathbf{c}^{[2]}\in\mathcal{C}_2]}, \tag{2.9}$$

which is the expression appearing in [6, Chapter 5]. Accordingly, for user 2 we have

$$\hat{x}_i^{[2]} = \arg\max_{x_i^{[2]}} \sum_{\sim x_i^{[2]}} \sum_{\mathbf{x}^{[1]}} \left(\prod_{k=1}^{n} p(y_k | x_k^{[1]}, x_k^{[2]}) \right) \mathbf{1}_{[\mathbf{c}^{[1]}\in\mathcal{C}_1]} \mathbf{1}_{[\mathbf{c}^{[2]}\in\mathcal{C}_2]}. \tag{2.10}$$

2.1.2 Two-User Tanner Graph and Tree Assumption

The above expressions naturally lend themselves to a Tanner graph representation. The BP algorithm [13] can be employed on this graph in order approximate (2.9) and (2.10) accurately and efficiently. In the ideal case where the Tanner graph is cycle-free, BP actually performs exact MAP decoding.

Since we have two users using possibly different codes, the two-user Tanner graph will contain two distinct single user Tanner graphs. Moreover, the function $p(y_k|x_k^{[1]}, x_k^{[2]})$ in (2.9) and (2.10) means that the variable node k of user 1 will be connected to the variable node k of user 2 through a node which represents this function. We call this node a *state node*. An example of the two-user Tanner graph which we just described is depicted in Figure 2.1.

The Tree Assumption for a single user graph states that, for a fixed number of iterations and in the limit of infinite blocklength, the BP computation graph, which is derived from the code's Tanner graph, is a tree, i.e. no cycles exist [6, Chapter 3]. In this case, the BP algorithm performs exact MAP decoding and the performance of a code ensemble can be predicted very elegantly. So, it is comforting to know that the following statement is true.

Theorem 2.1. *For a fixed number of iterations and in the limit of infinite blocklength for both codes, the BP computation graph, which is derived from the two-user Tanner graph, is cycle-free with probability one.*

Proof. The proof is very similar to the single-user case, see [6]. □

We will henceforth use the notions of the Tanner graph and the corresponding BP computation graph interchangeably, since each one can be derived from the other.

2.1.3 Belief Propagation

Now that we have asserted that in the limit of infinite blocklength the two-user Tanner graph is cycle-free, we can proceed with stating the message update rules for BP. All messages are in Log-Likelihood Ratio (LLR) form, i.e.

$$m = \log \frac{p(x_i = +1|y)}{p(x_i = -1|y)}, \tag{2.11}$$

where y is a random variable containing all the information incorporated into this message [14].

We will first discuss the update rules for variable and check nodes. Since these update rules are the same regardless of the user, we will temporarily suppress the user index for the sake of simplicity. Let sv_i^ℓ denote the state-to-variable message from state node i towards variable node i at iteration ℓ. Let cv_{ji}^ℓ denote the check-to-variable messages from check node j towards variable node i. The variable-to-check message from variable node i towards check node j at iteration $(\ell+1)$, denoted $vc_{ij}^{(\ell+1)}$, can be computed, according to standard BP rules [6], as

$$vc_{ij}^{(\ell+1)} = \begin{cases} sv_i^\ell, & \ell = 0, \\ sv_i^\ell + \sum_{j' \in C_i/\{j\}} cv_{j'i}^\ell, & \ell \geq 1, \end{cases} \tag{2.12}$$

where C_i denotes the set of all check nodes connected to variable node i. The variable-to-state message from variable node i towards state node i, denoted vs_i^ℓ, can be computed, according to standard BP rules [6], as

$$vs_i^\ell = \sum_{j' \in C_i} cv_{j'i}^\ell. \tag{2.13}$$

Both rules exhibit the following symmetry property

$$\Psi(-m_0, -m_1, \dots, -m_{|C_i/\{j\}|}) = -\Psi(m_0, m_1, \dots, -m_{|C_i/\{j\}|}), \tag{2.14}$$

where Ψ is the update rule appearing in Equation (2.12) or (2.13), and $m_0, \dots, m_{|C_i/\{j\}|}$ are the messages used in the calculation of Ψ. A hard decision for the BPSK codeword bit i at iteration ℓ can be made as follows

$$\hat{x}_i^\ell = \operatorname{sign}\left(sv_i^\ell + \sum_{j' \in C_i} cv_{j'i}^\ell \right). \tag{2.15}$$

The check-to-variable message from check node j towards variable node i can be computed, according to standard BP rules [6], as

$$cv_{ji}^{\ell} = \frac{1}{2}\tanh^{-1}\left(\prod_{i' \in V_j/\{i\}} \tanh\left(\frac{vc_{i'j}^{\ell}}{2}\right)\right), \tag{2.16}$$

where V_j denotes the set of all variable nodes connected to check node j. The check node update rule exhibits the following symmetry

$$\Phi(b_0 m_0, b_1 m_1, \ldots, b_{|V_j/\{i\}|} m_{|V_j/\{i\}|}) = \left(\prod_{k=0}^{|V_j/\{i\}|} b_k\right) \Phi(m_0, m_1, \ldots, m_{|V_j/\{i\}|}), \tag{2.17}$$

for any ± 1 sequence $(b_0, \ldots, b_{|V_j/\{i\}|})$, where Φ is the update rule appearing in Equation () and $m_0, \ldots, m_{|V_j/\{i\}|}$ are the messages used in the calculation of Φ. Until this point, we have reviewed the single user update rules for the BP messages and we have also introduced the variable-to-state message, which simply follows the same rule as the variable-to-check messages.

We now need to derive an update rule for the messages from the new type of nodes we introduced, i.e. the state nodes, towards the variable nodes. This can be achieved by applying the standard message passing rule from which the message update rule for the check nodes was derived [6], but with respect to the function which the state nodes represent, i.e. $p(y_i|x_i^{[1]}, x_i^{[2]})$. We will first consider the equal power case, deriving the results of [12] in detail. We will then move on to the unequal power case and derive the corresponding message update rule.

Equal Power: We focus on the derivation of the update rule for the messages towards the variable nodes of user 1. For user 2, the update rule can be derived accordingly. For the message from state node i towards variable node i of user 1, denoted $sv_i^{[1]}$, we use the variable-to-state messages from user 2 and the channel observation y_i. According to standard message passing rules [6], the outgoing message for state node i is

$$sv_i^{[1]} = \log \frac{\sum_{x_i^{[2]}} p(y_i|x_i^{[1]} = +1, x_i^{[2]})\mu(x_i^{[2]})}{\sum_{x_i^{[2]}} p(y_i|x_i^{[1]} = -1, x_i^{[2]})\mu(x_i^{[2]})}. \tag{2.18}$$

We note that $vs_i^{[2]} = \log \frac{\mu(x_i^{[2]}=+1)}{\mu(x_i^{[2]}=-1)}$. Since the update rule is the same for all state nodes, we will henceforth ommit the index i for the sake of simplicity. We have[1]

$$\begin{aligned} sv^{[1]} &= \log \frac{\sum_{x^{[2]}} p(y|+1, x^{[2]})\mu(x^{[2]})}{\sum_{x^{[2]}} p(y|-1, x^{[2]})\mu(x^{[2]})} && (2.19)\\ &= \log \frac{p(y|+1,+1)\mu^{[2]}(+1) + p(y|+1,-1)\mu^{[2]}(-1)}{p(y|-1,+1)\mu^{[2]}(+1) + p(y|-1,-1)\mu^{[2]}(-1)} && (2.20)\\ &= \log \frac{p(y|+1,+1)\frac{\mu^{[2]}(+1)}{\mu^{[2]}(-1)} + p(y|+1,-1)}{p(y|-1,+1)\frac{\mu^{[2]}(+1)}{\mu^{[2]}(-1)} + p(y|-1,-1)}. && (2.21) \end{aligned}$$

[1] For readability, we will use the notation $p(y|x^{[1]} = \pm 1, x^{[2]} = \pm 1) = p(y|\pm 1, \pm 1)$ and $\mu(x^{[2]} = \pm 1) = \mu^{[2]}(\pm 1)$.

As stated previously, we have $\frac{\mu^{[2]}(+1)}{\mu^{[2]}(-1)} = e^{vs^{[2]}}$, which leads to

$$
\begin{aligned}
sv^{[1]}(y, vs^{[2]}) &= \log \frac{\mathcal{N}(+2, \sigma^2)e^{vs^{[2]}} + \mathcal{N}(0, \sigma^2)}{\mathcal{N}(0, \sigma^2)e^{vs^{[2]}} + \mathcal{N}(-2, \sigma^2)} && (2.22)\\
&= \log \frac{\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y-2)^2}{2\sigma^2}} e^{vs^{[2]}} + \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{y^2}{2\sigma^2}}}{\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{y^2}{2\sigma^2}} e^{vs^{[2]}} + \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y+2)^2}{2\sigma^2}}} && (2.23)\\
&= \log \frac{e^{-\frac{(y-2)^2}{2\sigma^2}} e^{vs^{[2]}} e^{\frac{y^2}{2\sigma^2}} + 1}{e^{vs^{[2]}} + e^{-\frac{(y+2)^2}{2\sigma^2}} e^{-\frac{y^2}{2\sigma^2}}} && (2.24)\\
&= \log \frac{e^{\frac{2(y-1)}{\sigma^2} + vs^{[2]}} + 1}{e^{vs^{[2]}} + e^{-\frac{2(y+1)}{\sigma^2}}}, && (2.25)
\end{aligned}
$$

which leads to a final rule that is exactly the same as the one found in [12]. For completeness, the final state node update rules for the messages towards the variable nodes of both users are

$$
sv^{[1]}(y, vs^{[2]}) = \log \frac{e^{\frac{2(y-1)}{\sigma^2} + vs^{[2]}} + 1}{e^{vs^{[2]}} + e^{-\frac{2(y+1)}{\sigma^2}}}, \tag{2.26}
$$

$$
sv^{[2]}(y, vs^{[1]}) = \log \frac{e^{\frac{2(y-1)}{\sigma^2} + vs^{[1]}} + 1}{e^{vs^{[1]}} + e^{-\frac{2(y+1)}{\sigma^2}}}. \tag{2.27}
$$

Unequal Power: By working as we did in the previous section, but lifting the equal power assumption, for user 1 we have

$$
\begin{aligned}
sv^{[1]} &= \log \frac{p(y|x^{[1]} = +1)}{p(y|x^{[1]} = -1)} && (2.28)\\
&= \log \frac{p(y|+\sqrt{P_1}, +\sqrt{P_2})p(x^{[2]} = +\sqrt{P_2}) + p(y|+\sqrt{P_1}, -\sqrt{P_2})p(x^{[2]} = -\sqrt{P_2})}{p(y|-\sqrt{P_1}, +\sqrt{P_2})p(x^{[2]} = +\sqrt{P_2}) + p(y|-\sqrt{P_1}, -\sqrt{P_2})p(x^{[2]} = -\sqrt{P_2})} && (2.29)\\
&= \log \frac{\mathcal{N}(+\sqrt{P_1} + \sqrt{P_2}, \sigma^2)p(x^{[2]} = +\sqrt{P_2}) + \mathcal{N}(+\sqrt{P_1} - \sqrt{P_2}, \sigma^2)p(x^{[2]} = -\sqrt{P_2})}{\mathcal{N}(-\sqrt{P_1} + \sqrt{P_2}, \sigma^2)p(x^{[2]} = +\sqrt{P_2}) + \mathcal{N}(-\sqrt{P_1} - \sqrt{P_2}, \sigma^2)p(x^{[2]} = -\sqrt{P_2})} && (2.30)\\
&= \log \frac{\mathcal{N}(+\sqrt{P_1} + \sqrt{P_2}, \sigma^2)\frac{p(x^{[2]}=+1)}{p(x^{[2]}=-1)} + \mathcal{N}(+\sqrt{P_1} - \sqrt{P_2}, \sigma^2)}{\mathcal{N}(-\sqrt{P_1} + \sqrt{P_2}, \sigma^2)\frac{p(x^{[2]}=+1)}{p(x^{[2]}=-1)} + \mathcal{N}(-\sqrt{P_1} - \sqrt{P_2}, \sigma^2)} && (2.31)\\
&= \log \frac{\mathcal{N}(+\sqrt{P_1} + \sqrt{P_2}, \sigma^2)e^{vs^{[2]}} + \mathcal{N}(+\sqrt{P_1} - \sqrt{P_2}, \sigma^2)}{\mathcal{N}(-\sqrt{P_1} + \sqrt{P_2}, \sigma^2)e^{vs^{[2]}} + \mathcal{N}(-\sqrt{P_1} - \sqrt{P_2}, \sigma^2)}. && (2.32)
\end{aligned}
$$

From (2.28) to (2.29) we used the law of total probability. From (2.29) to (2.30) we used the conditional probability density function of the channel. From (2.30) to (2.31) we divided by $p(x^{[2]} = -1)$. If we replace the conditional channel probability density functions in (2.32) with their closed form expressions, we get the final update rule

$$
sv^{[1]}(y, vs^{[2]}) = \log \frac{e^{-\frac{(y-\sqrt{P_1}-\sqrt{P_2})^2}{2\sigma^2}} e^{vs^{[2]}} + e^{-\frac{(y-\sqrt{P_1}+\sqrt{P_2})^2}{2\sigma^2}}}{e^{-\frac{(y+\sqrt{P_1}-\sqrt{P_2})^2}{2\sigma^2}} e^{vs^{[2]}} + e^{-\frac{(y+\sqrt{P_1}+\sqrt{P_2})^2}{2\sigma^2}}}. \tag{2.33}
$$

We observe that this update rule boils down to the equal power update rule in (2.26) for the special case where $\sqrt{P_1} = \sqrt{P_2} = 1$. Recall that our model assumes $\sigma^2 = 1$, so for user 1 we get

$$sv^{[1]}(y, vs^{[2]}) = \log \frac{e^{-\frac{(y-\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{vs^{[2]}} + e^{-\frac{(y-\sqrt{P_1}+\sqrt{P_2})^2}{2}}}{e^{-\frac{(y+\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{vs^{[2]}} + e^{-\frac{(y+\sqrt{P_1}+\sqrt{P_2})^2}{2}}}. \tag{2.34}$$

Accordingly, for user, 2 we have

$$sv^{[2]}(y, vs^{[1]}) = \log \frac{e^{-\frac{(y-\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{vs^{[1]}} + e^{-\frac{(y+\sqrt{P_1}-\sqrt{P_2})^2}{2}}}{e^{-\frac{(y-\sqrt{P_1}+\sqrt{P_2})^2}{2}} e^{vs^{[1]}} + e^{-\frac{(y+\sqrt{P_1}+\sqrt{P_2})^2}{2}}}. \tag{2.35}$$

The slight difference of the signs in (2.34) and (2.35) is the only indication of the asymmetry of the update rules of the two users.

2.1.4 Message Scheduling

After deriving the message update rules for all types of nodes, in order to completely describe the decoder, we need to specify a schedule, i.e. an order in which the messages will be computed. We choose a parallel schedule in which the state node messages are first updated and then a round of single user BP is performed on each single user Tanner graph. This schedule admits easier analysis than most other schedules [6, Chapter 5],[12]. We consider one iteration to start with a state-to-variable message update, followed by the single user variable-to-check and check-to-variable message updates and ending with the variable-to-state updates. No claims for the optimality of this schedule are made.

2.2 Density Evolution

Density Evolution (DE) tracks the probability density functions of the messages exchanged by the BP decoder during the ℓ-th iteration, for $\ell = 1, 2, \ldots$. Using these densities, or some linear functional of the densities (like the Bit Error Rate (BER) or the entropy), we can derive conditions which guarantee that the probability of decoding error becomes vanishingly small as the number of iterations tends to infinity.

In this section, we first prove that, in order to predict the asymptotic performance of a pair of LDPC code ensembles over the two-user GMAC, it suffices to consider the case where one user transmits the all-zero codeword (equivalently, the all-one BPSK codeword) and the other user transmits a typical codeword of type one-half, i.e. a codeword with half its bits equal to zero (equivalently, half its symbols equal to $+1$) and half its bits equal to one (equivalently, half its symbols equal to -1). We then present the recursive two-user DE update rules.

2.2.1 (No) Restriction to the All-One BPSK Codeword

The GMAC is not symmetric in the single user sense. Due to this fact, the probability of error is not codeword independent, meaning that we can not restrict our analysis to the case where both users transmit the all-one BPSK codeword. Fortunately, we can show that it suffices to consider the case where one user transmits the all-one BPSK codeword and the other user transmits a BPSK codeword with half positions equal to $+1$ and the other half equal to -1. In order to provide intuition for the proof of the aforementioned statement and to introduce some notation, we will first state the proof for the restriction to the all-one BPSK codeword in the single user case.

Recall that the Log-Likelihood Ratio for the noisy output y of a binary memoryless channel is defined as

$$l(y) = \log \frac{p_{Y|X}(y|x=+1)}{p_{Y|X}(y|x=-1)}. \tag{2.36}$$

In the single user case, a binary memoryless channel is said to be *symmetric* if

$$p_{Y|X}(y|x=+1) = p_{Y|X}(-y|x=-1). \tag{2.37}$$

Let Y denote the random variable corresponding to y and let $L(Y)$ denote the corresponding LLR. Let a denote the probability density function of $L(Y)$, conditioned on $X=1$. We say that a is an *L-density.* An L-density is *symmetric* if [6]

$$a(z) = e^x a(-z), \tag{2.38}$$

for all $z \in \mathbb{R}$. L-densities for binary memoryless symmetric channels are always symmetric [6]. L-densities can be used in order to represent the associated channel multiplicatively as

$$Y = xZ, \quad Z \sim a(z). \tag{2.39}$$

Theorem 2.2 ([6], Lemma 4.90)**.** *Let G be a Tanner graph representing a binary linear code $\mathcal{C}$. Suppose that $\mathcal{C}$ is used to transmit over a BMS channel characterized by its L-density a_{BMSC} and suppose that the receiver performs belief propagation decoding on G. Let $P^{BP}(G, a_{BMSC}, \ell, \mathbf{c})$ denote the conditional (bit or block) probability of error after the ℓ-th decoding iteration, assuming that $\mathbf{c}$ was sent, $\mathbf{c} \in \mathcal{C}$. Then, it holds that*

$$P^{BP}(G, a_{BMSC}, \ell, \mathbf{c}) = \frac{1}{|\mathcal{C}|} \sum_{\mathbf{c}' \in \mathcal{C}} P^{BP}(G, a_{BMSC}, \ell, \mathbf{c}') = P^{BP}(G, a_{BMSC}, \ell). \tag{2.40}$$

This means that $P^{BP}(G, a_{BMSC}, \ell, \mathbf{c})$ is independent of the transmitted codeword.

Proof. Recall that, based on the previous discussion, a binary memoryless symmetric channel can be modeled multiplicatively as

$$Y_t = x_t Z_t, \tag{2.41}$$

where $\{Z_t\}_{t=1}^n$ is a sequence of iid random variables with density a_{BMSC}. Let $\mathbf{x} \in \mathcal{C}$ and let $\mathbf{Y}$ denote the corresponding channel output, $\mathbf{Y} = \mathbf{xZ}$ (multiplication is component-wise). Note that $\mathbf{Z}$ by itself is equal to the observation assuming the the all-one codeword was transmitted. We will now show that the messages sent during the decoding process for the cases where the received word is either $\mathbf{xZ}$ or $\mathbf{Z}$ are in one-to-one correspondence.

Let i be an arbitrary variable node and let j be one of its neighboring check nodes. Let $m_{ij}^{(\ell)}(\mathbf{y})$ denote the message sent from i to j in iteration ℓ assuming that the received value is $\mathbf{y}$ and let $m_{ji}^{(\ell)}(\mathbf{y})$ denote the corresponding message sent from j to i.

We have $m_{ij}^{(0)}(\mathbf{y}) = m_{ij}^{(0)}(\mathbf{xz}) = x_i m_{ij}^{(0)}(\mathbf{z})$, where the second step follows from the variable node symmetry property in Equation (2.14). Assume that we have $m_{ij}^{(\ell)}(\mathbf{y}) = x_i m_{ij}^{(\ell)}(\mathbf{z})$ for all (i,j) pairs and some $\ell \geq 0$. Let ∂j denote all variable nodes which are connected to check node j. Since $\mathbf{x}$ is a codeword, we have

$$\prod_{k \in \partial j} x_k = 1 \quad \Leftrightarrow \quad \prod_{k \in \partial j / i} x_k = x_i. \tag{2.42}$$

In words, the product of the signs of the messages which are used for the calculation of the message towards variable node i is equal to x_i. Thus, from the check node symmetry condition in Equation (2.17), we conclude that $m_{ji}^{(\ell+1)}(\mathbf{y}) = x_i m_{ji}^{(\ell+1)}(\mathbf{z})$. Further, invoking once more the variable node symmetry condition, it follows that $m_{ij}^{(\ell+1)}(\mathbf{y}) = x_i m_{ij}^{(\ell+1)}(\mathbf{z})$ for all (i,j) pairs. Thus, by induction, all messages to and from variable node i, when $\mathbf{y}$ is received, are equal to the product of x_i and the corresponding message when $\mathbf{z}$ is received. Hence, both decoders proceed in lock step and commit exactly the same number of errors, which proves the claim. $\square$

We will now show that the state node update rule has a certain symmetry, which will be indispensable in the sequel.

Lemma 2.3. *Regardless of the users' powers, the update rules for the state-to-variable messages exhibit the following symmetry property*

$$sv^{[1]}(-y, -vs^{[2]}) = -sv^{[1]}(y, vs^{[2]}), \tag{2.43}$$

$$sv^{[2]}(-y, -vs^{[1]}) = -sv^{[2]}(y, vs^{[1]}). \tag{2.44}$$

Proof. For the equal power update rule for the messages towards user 1, it holds that [12]

$$sv^{[1]}(-y, -vs^{[2]}) = \log \frac{e^{\frac{2(-y-1)}{\sigma^2} - sv^{[2]}} + 1}{e^{-sv^{[2]}} + e^{-\frac{2(-y+1)}{\sigma^2}}} = \log \frac{e^{-\frac{2(y+1)}{\sigma^2} - sv^{[2]}} + 1}{e^{-sv^{[2]}} + e^{\frac{2(y-1)}{\sigma^2}}} \tag{2.45}$$

$$= -\log \frac{e^{\frac{2(y-1)}{\sigma^2} + sv^{[2]}} + 1}{e^{sv^{[2]}} + e^{-\frac{2(y+1)}{\sigma^2}}} = -sv^{[1]}(y, vs^{[2]}). \tag{2.46}$$

The state node update rule for the message towards user 2 exhibits the same symmetry. For the unequal power update rule for the messages towards user 1, which we derived in (2.34), it holds that

$$sv^{[1]}(-y, -vs^{[2]}) = \log \frac{e^{-\frac{(-y-\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{-vs^{[2]}} + e^{-\frac{(-y-\sqrt{P_1}+\sqrt{P_2})^2}{2}}}{e^{-\frac{(-y+\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{-vs^{[2]}} + e^{-\frac{(-y+\sqrt{P_1}+\sqrt{P_2})^2}{2}}} \tag{2.47}$$

$$= \log \frac{e^{-\frac{(y+\sqrt{P_1}+\sqrt{P_2})^2}{2}} e^{-vs^{[2]}} + e^{-\frac{(y+\sqrt{P_1}-\sqrt{P_2})^2}{2}}}{e^{-\frac{(y-\sqrt{P_1}+\sqrt{P_2})^2}{2}} e^{-vs^{[2]}} + e^{-\frac{(y-\sqrt{P_1}-\sqrt{P_2})^2}{2}}} \tag{2.48}$$

$$= \log \frac{e^{-\frac{(y+\sqrt{P_1}+\sqrt{P_2})^2}{2}} + e^{-\frac{(y+\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{vs^{[2]}}}{e^{-\frac{(y-\sqrt{P_1}+\sqrt{P_2})^2}{2}} + e^{-\frac{(y-\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{vs^{[2]}}} \tag{2.49}$$

$$= -sv^{[1]}(y, vs^{[2]}). \tag{2.50}$$

Again, the state node update rule for the message towards user 2 exhibits the same symmetry. $\square$

The following symmetry property of the conditional probabilities of the GMAC can be easily verified for the unequal power case

$$p(y_i | x_i^{[1]} = 1, x_i^{[2]} = 1) = p(-y_i | x_i^{[1]} = -1, x_i^{[2]} = -1). \tag{2.51}$$

Naturally, it also holds under the equal power assumption as a special case. If we restrict ourselves to the cases where $x_i^{[1]} = x_i^{[2]}$, this channel is equivalent to a BI-AWGN channel with input power $\sqrt{P_1} + \sqrt{P_2}$, which is symmetric in the single user sense. Similarly, if we restrict ourselves to the cases where $x_i^{[1]} = -x_i^{[2]}$, this channel is equivalent to a BI-AWGN channel with input power $\sqrt{P_1} - \sqrt{P_2}$, which is also symmetric in the single user sense. So, we can model transmission over the two "restricted" channels multiplicatively as [6]

$$Y_i^{[1]} = x_i Z_i^{[1]} \quad \text{and} \quad Y_i^{[2]} = x_i Z_i^{[2]}, \tag{2.52}$$

where $Z_i^{[1]}$ is distributed according to $a_{x_i^{[2]}=x_i^{[2]}}(z)$ and $Z_i^{[2]}$ is distributed according to $a_{x_i^{[1]}=-x_i^{[2]}}(z)$, and $x_i = x_i^{[1]} = x_i^{[2]}$ and $x_i = x_i^{[1]} = -x_i^{[2]}$, respectively.

Since $\mathbf{x}^{[1]}$ and $\mathbf{x}^{[2]}$ are independent, the events $x_i^{[2]} = x_i^{[2]}$ and $x_i^{[1]} = -x_i^{[2]}$ are equiprobable, each having probability 1/2. Thus, in the typical case half the state nodes will be $x_i^{[1]} = x_i^{[2]}$ nodes and half the state nodes will be $x_i^{[1]} = -x_i^{[2]}$ nodes.

Theorem 2.4. *The probability of error when the received values at $x_i^{[1]} = x_i^{[2]}$ nodes are $\mathbf{Y}^{[1]} = \mathbf{x}\mathbf{Z}^{[1]}$ and the received values at $x_i^{[1]} = -x_i^{[2]}$ nodes are $\mathbf{Y}^{[2]} = \mathbf{x}\mathbf{Z}^{[2]}$ is equal to the probability of error when the received values at $x_i^{[1]} = x_i^{[2]}$ nodes are $\mathbf{Y}^{[1]} = \mathbf{Z}^{[1]}$ and the received values at $x_i^{[1]} = -x_i^{[2]}$ nodes are $\mathbf{Y}^{[2]} = \mathbf{Z}^{[2]}$. In words, we can predict the asymptotic performance by assuming that user* 1 *transmits the all-one codeword and user* 2 *transmits a typical codeword of type one-half.*

Proof Outline: We will show that, due to the BP update rule symmetries, the x_i signs can be factored out. This means that we can assume transmission of the all-ones $\mathbf{x}$ codeword. The fact that, the probability of the event $x_i = x_i^{[1]} = x_i^{[2]}$ is 1/2 and the probability of the event $x_i = x_i^{[1]} = -x_i^{[2]}$ is also 1/2, leads to the conclusion that user 1 can transmit the all-one codeword, but user 2 has to transmit a codeword of type one-half.

Proof. We will prove Theorem 2.4 by induction. Let i_k be an arbitrary variable node of user $k,\ k = 1, 2$, and let j_k be one of its neighboring check nodes. Let $vc_{i_k j_k}^{(\ell)}(y_i)$ denote the message sent from i_k to j_k in iteration ℓ assuming that the received value is y_i, let $cv_{j_k i_k}^{(\ell)}(y_i)$ denote the corresponding message sent from j_k to i_k, and let $vs_{i_k}^{(\ell)}(y_i)$ denote the message sent from variable node i_k to its corresponding state node. Finally, let $sv_{i_2}^{(\ell)}(y, vs_{i_1}^{(\ell)})$ (resp. $sv_{i_1}^{(\ell)}(y, vs_{i_2}^{(\ell)})$) denote the message from the state node connected to variable node i_1 of user 1 (resp. variable node i_2 of user 2) towards the corresponding variable node of user 2, i.e. i_2 (resp. i_1).

Induction Basis: Due to the state node update rule symmetry, the initial messages from the state nodes to the variable nodes are

$$sv_{i_1}^{(0)}(y_i, 0) = sv_{i_1}^{(0)}(x_i z_i, 0) = x_i sv_{i_1}^{(0)}(z_i, 0), \tag{2.53}$$

$$sv_{i_2}^{(0)}(y_i, 0) = sv_{i_2}^{(0)}(x_i z_i, 0) = x_i sv_{i_2}^{(0)}(z_i, 0), \tag{2.54}$$

where z_i is distributed either according to $a_{x_i^{[1]}=x_i^{[2]}}(z)$ or according to $a_{x_i^{[1]}=-x_i^{[2]}}(z)$, depending on the type of the state node. Due to the variable node update rule symmetry and the fact the we can factor out the sign from the initial messages from the state nodes, we also have

$$vc_{i_k j_k}^{(0)}(y_i) = vc_{i_k j_k}^{(0)}(x_i z_i) = x_i vc_{i_k j_k}^{(0)}(z_i), \tag{2.55}$$

where z_i is distributed according either to $a_{x_i^{[1]}=x_i^{[2]}}(z)$ or to $a_{x_i^{[1]}=-x_i^{[2]}}(z)$, depending on the type of the state node the variable node is connected to. In either case, the sign can be factored out.

Inductive Hypothesis: Now assume that we have $vc_{i_k j_k}^{(\ell)}(y_i) = x_i vc_{i_k j_k}^{(\ell)}(z_i)$ for all (i_k, j_k) pairs.

Inductive Step: Let ∂j_k denote the indices of all variable nodes which are connected to check node j_k. Since $\mathbf{x_1}$ and $\mathbf{x_2}$ are codewords, they must satisfy all parity check equations. Thus, we have

$$\prod_{i \in \partial j_k} x_i = 1 \quad \Longrightarrow \quad \prod_{i \in \partial j_k / \{i'\}} x_i = x_{i'}, \tag{2.56}$$

where $i' \in \partial j_k$. So, using the check node symmetry, we have

$$cv_{j_k i_k}^{(\ell+1)}(y_i) = x_i cv_{j_k i_k}^{(\ell+1)}(z_i). \tag{2.57}$$

Due to the variable node update rule symmetry, which also holds for the messages from variable nodes to state nodes, we have

$$vs_{i_k}^{(\ell+1)}(y_i) = x_i vs_{i_k}^{(\ell+1)}(z_i). \tag{2.58}$$

Due to the state node update rule symmetry in Equation (2.44), for the message from the state node connected to variable node i_1 towards the corresponding variable node of user 2, we have

$$sv_{i_2}^{(\ell+1)}(x_i z_i, x_i vs_{i_1}^{(\ell+1)}) = x_i sv_{i_1}^{(\ell+1)}(z_i, vs_{i_1}^{(\ell+1)}). \tag{2.59}$$

Similarly, due to the state node update rule symmetry in Equation (2.43), for the message from the state node connected to variable node i_2 towards the corresponding variable node of user 1, we have

$$sv_{i_1}^{(\ell+1)}(x_i z_i, x_i vs_{i_2}^{(\ell+1)}) = x_i sv_{i_1}^{(\ell+1)}(z_i, vs_{i_2}^{(\ell+1)}). \tag{2.60}$$

Invoking the variable node symmetry once again, we have that

$$vc_{i_k j_k}^{(\ell+1)}(y_i) = x_i vc_{i_k j_k}^{(\ell+1)}(z_i). \tag{2.61}$$

Thus, we have proven the inductive step. This means that $\mathbf{x}$ can be the all-zero codeword, which in turn implies that $\mathbf{x}^{[1]}$ can be the all-zero codeword but $\mathbf{x}^{[2]}$ has to be a codeword of type one-half. $\square$

Remark. If we set $x_i = -x_i^{[1]} = x_i^{[2]}$ in (2.52) when $x_i^{[1]} = -x_i^{[2]}$ (instead of $x_i = x_i^{[1]} = -x_i^{[2]}$ as we did), then nothing changes essentially except that we will reach the conclusion that user 2 can be restricted to the all-one codeword and user 1 has to transmit a typical codeword of type one-half. In both cases, one user transmits the all-one codeword and the other user transmits a codeword of type one-half.

2.2.2 Density Evolution

Before continuing, we need to write the check node message update rule in a more convenient form. Let γ denote a map from $[-\infty, +\infty]$ to $\mathrm{GF}(2) \times [0, +\infty]$ defined as [14]

$$\gamma(x) \triangleq (\gamma_1(x), \gamma_2(x)) \triangleq \left(\operatorname{sign}(x), -\ln\tanh\left|\frac{x}{2}\right|\right), \tag{2.62}$$

where the convention $-\ln 0 = +\infty$ is used. Using $\gamma(x)$, the check node message update rule in (2.1.3) can be rewritten as

$$cv_{ji}^{\ell} = \gamma^{-1}\left(\sum_{i' \in V_j/\{i\}} \gamma\left(vc_{i'j}^{\ell}\right)\right), \tag{2.63}$$

where addition is performed component-wise. This form is more convenient because, using the Tree Assumption and given the densities of all $vc_{i'j}^{\ell}$, $i' \in V_j$, it is relatively easy to calculate the density of cv_{ji}^{ℓ}. Let Γ denote the density transformation corresponding to $\gamma(x)$, i.e. given a random variable $Z \in [-\infty, +\infty]$ with density $f_Z(z)$, the density of $\gamma(Z)$ is

$$f_{\gamma(Z)}(\gamma_1(z), \gamma_2(z)) = \Gamma(f_Z(z)). \tag{2.64}$$

This transformation has a well defined inverse, denoted Γ^{-1}, and both Γ and Γ^{-1} are additive operators on their respective domain spaces [14]. The range space of Γ is endowed with a convolution operator [14].

As in [14], let P_ℓ and Q_ℓ denote the densities of the random variables vc^ℓ and cv^ℓ, respectively. We wish to calculate the density of the messages emanating from a check node of degree i, denoted $Q_{\ell,i}$. At a check node of degree i, $(i-1)$ incoming messages[2] are summed after passing through the transformation $\gamma(x)$. The result is then transformed back to the LLR domain using γ^{-1}. By the Tree Assumption, all incoming messages are independent and distributed according to P_ℓ and their images under $\gamma(x)$ are also independent and distributed according to $\Gamma(P_\ell)$. This means that the density of the sum in (2.63) is equal to the $(i-1)$-fold convolution of $\Gamma(P_\ell)$ with itself over the space defined by Γ. The last step is to apply the Γ^{-1} transformation to the resulting density, leading to

$$Q_{\ell,i} = \Gamma^{-1}\left(\Gamma\left(P_\ell\right)^{\otimes(i-1)}\right). \tag{2.65}$$

By definition of $\rho(x)$, an edge is connected to a check node of degree i with probability ρ_i. Thus, the above density corresponds to a message that appears in BP decoding with probability ρ_i. By taking the weighted average of the densities $Q_{\ell,i}$ over the coefficients of $\rho(x)$, we have

$$Q_\ell = \sum_i \rho_i Q_{\ell,i} = \sum_i \rho_i \Gamma^{-1}\left(\Gamma\left(P_\ell\right)^{\otimes(i-1)}\right). \tag{2.66}$$

Γ^{-1} is additive, thus

$$\sum_i \rho_i \Gamma^{-1}\left(\Gamma\left(P_\ell\right)^{\otimes(i-1)}\right) = \Gamma^{-1}\left(\sum_i \rho_i \Gamma\left(P_\ell\right)^{\otimes(i-1)}\right), \tag{2.67}$$

which we can rewrite, slightly abusing notation, to get the final density update rule for the check nodes

$$Q_\ell = \Gamma^{-1}\left(\rho\left(\Gamma\left(P_\ell\right)\right)\right). \tag{2.68}$$

At variable nodes the situation is much less involved. At a degree i variable node, $(i-1)$ incoming messages from check nodes are summed along with an additional incoming message. Let S_ℓ denote the density of this additional incoming message at iteration ℓ. By the Tree Assumption, the messages from check nodes are independent and identically distributed according

[2]Recall that these messages are LLRs.

to Q_ℓ and the additional message is independent from all other incoming messages and distributed according to S_ℓ. Then, the density of the messages emanating from a variable node of degree i towards the check nodes, denoted $P_{(\ell+1),i}$, is simply the convolution of the densities of all incoming messages, i.e.

$$P_{(\ell+1),i} = S_\ell \otimes Q_\ell^{\otimes(i-1)}. \tag{2.69}$$

By averaging over the coefficients of $\lambda(x)$ and by substituting the expression we have derived for Q_ℓ as a function of P_ℓ, we have

$$\begin{aligned} P_{(\ell+1)} &= \sum_i \lambda_i \left(S_\ell \otimes Q_\ell^{\otimes(i-1)} \right) && (2.70)\\ &= S_\ell \otimes \sum_i \lambda_i Q_\ell^{\otimes(i-1)} && (2.71)\\ &= S_\ell \otimes \sum_i \lambda_i \left(\Gamma^{-1} \left(\rho \left(\Gamma \left(P_\ell \right) \right) \right) \right)^{\otimes(i-1)}. && (2.72) \end{aligned}$$

By slightly abusing notation again, we get the final expression for $P_{(\ell+1)}$ as a function of P_ℓ

$$P_{(\ell+1)} = S_\ell \otimes \lambda \left(\Gamma^{-1} \left(\rho \left(\Gamma \left(P_\ell \right) \right) \right) \right). \tag{2.73}$$

In single user Density Evolution, S_ℓ depends solely on the channel observation and it is actually not a function of ℓ. In the two-user GMAC, however, this density corresponds to incoming messages from the state nodes. Since messages from the state nodes towards one user depend on the check-to-variable messages of the other user, the recursive DE expression for each user will necessarily contain some message density of the other user.

Let R_ℓ denote density of the variable-to-state messages at iteration ℓ. The message from a degree i variable node towards its corresponding state node is the sum of i independent incoming messages from the check nodes which are distributed according to Q_ℓ. So, the density of the message from a degree i variable node towards its state node at iteration ℓ, denoted $R_{\ell,i}$, is the convolution of the densities of the incoming messages, i.e.

$$R_{\ell,i} = Q_\ell^{\otimes i}. \tag{2.74}$$

The densities $R_{\ell,i}$ depend on the degree of the variable node from which the corresponding messages emanate, not from the degree of the edge along which they are transmitted, as happens with variable-to-check messages. Thus, in order to calculate the average density R_ℓ, we need to average $R_{\ell,i}$ over the degrees of the variable nodes,[3] i.e. over the variable node degree distribution from a node perspective, which is denoted as $L(x)$. So, we have

$$R_\ell = \sum_i L_i R_{\ell,i} = \sum_i L_i Q_\ell^{\otimes i} = \sum_i L(Q_\ell), \tag{2.75}$$

where we have slightly abused notation as usual.

Let Σ denote the density transformation corresponding to the state-to-variable update rule. Then, resuming user notation, we can express $P_{(\ell+1)}^{[1]}$ as a function of both $P_\ell^{[1]}$ and $P_\ell^{[2]}$ as

[3]Recall that for P_ℓ and Q_ℓ we averaged over the degree of the edges, not the degrees of the nodes.

follows

$$P^{[1]}_{(\ell+1)} = S^{[2]}_{\ell} \otimes \lambda^{[1]}\left(\Gamma^{-1}\left(\rho^{[1]}\left(\Gamma\left(P^{[1]}_{\ell}\right)\right)\right)\right) \tag{2.76}$$

$$= \Sigma\left(R^{[2]}_{\ell}\right) \otimes \lambda^{[1]}\left(\Gamma^{-1}\left(\rho^{[1]}\left(\Gamma\left(P^{[1]}_{\ell}\right)\right)\right)\right) \tag{2.77}$$

$$= \Sigma\left(L^{[2]}\left(Q^{[2]}_{\ell}\right)\right) \otimes \lambda^{[1]}\left(\Gamma^{-1}\left(\rho^{[1]}\left(\Gamma\left(P^{[1]}_{\ell}\right)\right)\right)\right) \tag{2.78}$$

$$= \Sigma\left(L^{[2]}\left(\Gamma^{-1}\left(\rho^{[2]}\left(\Gamma\left(P^{[2]}_{\ell}\right)\right)\right)\right)\right) \otimes \lambda^{[1]}\left(\Gamma^{-1}\left(\rho^{[1]}\left(\Gamma\left(P^{[1]}_{\ell}\right)\right)\right)\right). \tag{2.79}$$

Similarly, for $P^{[2]}_{(\ell+1)}$ we have

$$P^{[2]}_{(\ell+1)} = \Sigma\left(L^{[1]}\left(\Gamma^{-1}\left(\rho^{[1]}\left(\Gamma\left(P^{[1]}_{\ell}\right)\right)\right)\right)\right) \otimes \lambda^{[2]}\left(\Gamma^{-1}\left(\rho^{[2]}\left(\Gamma\left(P^{[2]}_{\ell}\right)\right)\right)\right). \tag{2.80}$$

The derivation of Σ is nontrivial. However, we can overcome this problem by proceeding with a Gaussian Approximation, in which the exact nature of Σ is irrelevant.

2.2.3 Stability Condition

Before proceeding with the Gaussian Approximation, we will very briefly discuss the stability condition for the two-user Density Evolution recursions. The stability condition provides a necessary and sufficient condition for the decoding to converge to zero probability of error, provided that the probability of error has already evolved to a sufficiently small value [14].

For the equal power case, we use the stability condition which is provided in [12]. The same stability condition applies to both users since in [12] it is assumed that both users employ codes from the same ensemble.

$$\lambda_2 \sum_i (i-1)\rho_i < \exp\left(1/(2\sigma^2)\right). \tag{2.81}$$

We observe that the receiver of an unequal power two-user GMAC channel is equivalent to the best receiver of a two-user broadcast channel. Thus, the stability condition of the unequal power two-user DE recursions decomposes into to two single user stability conditions, as in [15]

$$\lambda^{[j]}_2 \sum_i (i-1)\rho^{[j]}_i < \exp\left(P_j/2\right), \; j = 1, 2. \tag{2.82}$$

2.3 Gaussian Approximation and EXIT Charts

In this section, we apply the Gaussian Approximation (GA) to the two-user Density Evolution recursion and provide conditions which guarantee that the probability of decoding error for both users becomes vanishingly small. Based on the GA, we proceed with an EXtrinsic Information Transfer (EXIT) chart analysis and we provide EXIT chart based sufficient conditions for successful decoding.

2.3.1 Gaussian Approximation

Under the Gaussian Approximation [16], all message densities are approximated as symmetric Gaussian. A Gaussian density is said to be symmetric if $\sigma^2 = 2\mu$, where σ^2 and μ denote the variance and the mean, respectively. So, instead of tracking the evolution of densities as the number of iterations increases, we can get a very good approximation of DE by tracking the evolution of a single parameter, i.e. the mean.

We will temporarily drop the user index for the sake of simplicity when dealing with the variable and check nodes, since the same rules apply to both users.

Let μ_{cv}^{ℓ} denote the mean of the check-to-variable messages at iteration ℓ. Similarly, let μ_{sv}^{ℓ} denote the mean of the state-to-variable messages at iteration ℓ. Then, the mean of the variable-to-check messages emanating from a variable node of degree i at iteration $(\ell+1)$, denoted $\mu_{vc}^{(\ell+1),i}$, is

$$\mu_{vc}^{(\ell+1),i} = \mu_{sv}^{\ell} + (i-1)\mu_{cv}^{\ell}. \tag{2.83}$$

At a variable node of degree i, due to the central limit theorem and the fact that all incoming messages are assumed to be independent, the density of the variable-to-check message does indeed resemble a Gaussian density. As the degree of the variable node increases, the approximation becomes better. By averaging over the coefficients of $\lambda(x)$, the overall variable-to-check message density will be a mixture of Gaussians with mean

$$\mu_{vc}^{(\ell+1)} = \sum_i \lambda_i \mu_{vc}^{(\ell+1),i} = \mu_{sv}^{\ell} + \sum_i \lambda_i (i-1)\mu_{cv}^{\ell}. \tag{2.84}$$

However, under the GA, it is assumed that the above density is not a mixture of Gaussian densities, but a single Gaussian density with mean $\mu_{vc}^{(\ell+1)}$ [16].

The mean of the variable-to-state messages emanating from a variable node of degree i at iteration ℓ, denoted $\mu_{vs}^{(\ell),i}$, is

$$\mu_{vs}^{\ell,i} = i\mu_{cv}^{\ell}. \tag{2.85}$$

Since we average over the coefficients of $L(x)$, the overall variable-to-state message density will also be a mixture of Gaussians with mean

$$\mu_{vs}^{\ell} = \sum_i L_i \mu_{vs}^{(\ell),i} = \sum_i L_i i \mu_{cv}^{\ell}. \tag{2.86}$$

As previously, under the GA it is assumed that the above density is not a mixture of Gaussians, but a single Gaussian density with mean μ_{vs}^{ℓ}.

At check nodes the approximation is not very accurate, but it has been shown to be satisfactory [16]. As before, since we average over the coefficients of $\rho(x)$, the actual overall density will be a mixture of Gaussians, but we assume that it is a single Gaussian density. As shown in [16], at a degree i check node we have

$$\mu_{cv}^{\ell,i} = \phi^{-1}\left(1 - \left(1 - \phi(\mu_{vc}^{\ell})\right)^{(i-1)}\right), \tag{2.87}$$

where $\phi(\cdot)$ is defined as

$$\phi(x) = \begin{cases} 1 - \frac{1}{\sqrt{4\pi x}} \int_{-\infty}^{+\infty} \tanh\left(\frac{u}{2}\right) \exp\left\{-\frac{(u-x)^2}{4x}\right\} du, & x > 0, \\ 1, & x = 0. \end{cases} \tag{2.88}$$

A good approximation for $\phi(x)$ can be found in Appendix C. By averaging over the coefficients of $\rho(x)$, we get

$$\mu_{cv}^{\ell} = \sum_i \rho_i \mu_{cv}^{\ell,i} = \sum_i \rho_i \phi^{-1}\left(1 - \left(1 - \phi(\mu_{vc}^{\ell})\right)^{(i-1)}\right). \tag{2.89}$$

By combining (2.84) and (2.89), we can express $\mu_{vc}^{(\ell+1)}$ as a function of $\mu_{vc}^{(\ell)}$ as follows

$$\mu_{vc}^{(\ell+1)} = \mu_{sv}^{\ell} + \sum_i \lambda_i (i-1)\left(\sum_j \rho_j \phi^{-1}\left(1 - \left(1 - \phi(\mu_{vc}^{\ell})\right)^{(j-1)}\right)\right). \tag{2.90}$$

As a last step, we need to derive the update rule for the state-to-variable message means. At this point, we will resume the user index notation since it is necessary. We state the equal power update rule appearing in [12] and we derive a new update rule for the unequal power case.

As discussed previously, we can restrict the transmitted codeword to the all-one BPSK codeword for one user and a type one-half BPSK codeword for the other user. This means that we will have two types of state nodes. One type is connected to two variable nodes with associated value $+1$, while the other is connected to one variable node with associated value $+1$ and one variable node with associated value -1.

Equal Power: Recall that the state node update rule for user 1 is

$$sv^{[1],\ell}(y, vs^{[2],\ell}) = \log \frac{e^{\frac{2(y-1)}{\sigma^2}+vs^{[2],\ell}} + 1}{e^{vs^{[2],\ell}} + e^{-\frac{2(y+1)}{\sigma^2}}}. \tag{2.91}$$

We assume that the variable-to-state message from user 2, denoted $vs^{[2],\ell}$, has mean $\mu_{vs}^{[2],\ell}$. The mean of y depends on the type of the state node. Using the above update rule and given the means of the input messages, for a state node of user 1 which is connected to two $+1$ variable nodes, the mean of the state-to-variable messages towards user 2 at iteration ℓ can be calculated as

$$\mu_{sv|+1,+1}^{[1],\ell} = F_{+1,+1}(\mu_{vs}^{[2],\ell}, \sigma^2), \tag{2.92}$$

where σ^2 is the noise variance and $F_{+1,+1}(\mu, \sigma^2)$ is defined as [12][4]

$$F_{+1,+1}(\mu, \sigma^2) \triangleq \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log \left(\frac{1 + e^{2z\sqrt{\mu+2/\sigma^2}+\mu+2/\sigma^2}}{1 + e^{-2z\sqrt{\mu+2/\sigma^2}-\mu-6/\sigma^2}} \right) dz - \mu. \tag{2.93}$$

Similarly, for the state nodes connected to one $+1$ and one -1 variable node, the mean of the state-to-variable messages towards user 2 at iteration ℓ can be calculated as

$$\mu_{sv|+1,-1}^{[1],\ell} = F_{+1,-1}(\mu_{vs}^{[2],\ell}, \sigma^2), \tag{2.94}$$

where $F_{+1,-1}(\mu, \sigma^2)$ is defined as [12][4]

$$F_{+1,-1}(\mu, \sigma^2) \triangleq \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log \left(\frac{1 + e^{2z\sqrt{\mu+2/\sigma^2}-\mu-2/\sigma^2}}{1 + e^{-2z\sqrt{\mu+2/\sigma^2}+\mu-2/\sigma^2}} \right) dz + \mu. \tag{2.95}$$

Using the law of iterated expectations, the overall mean can be calculated as

$$\begin{aligned}
\mu_{sv}^{[1],\ell} &= p(+1,+1) \cdot \mu_{sv|+1,+1}^{[1],\ell} + p(+1,-1) \cdot \mu_{sv|+1,-1}^{[1],\ell} && (2.96)\\
&= p(+1,+1) \cdot F_{+1,+1}(\mu_{vs}^{[2],\ell}, \sigma^2) + p(+1,-1) \cdot F_{+1,-1}(\mu_{vs}^{[2],\ell}, \sigma^2) && (2.97)\\
&= \frac{1}{2} \left(F_{+1,+1}(\mu_{vs}^{[2],\ell}, \sigma^2) + F_{+1,-1}(\mu_{vs}^{[2],\ell}, \sigma^2) \right). && (2.98)
\end{aligned}$$

For user 2, due to symmetry, we have

$$\mu_{sv}^{[2],\ell} = \frac{1}{2} \left(F_{+1,+1}(\mu_{vs}^{[1],\ell}, \sigma^2) + F_{+1,-1}(\mu_{vs}^{[1],\ell}, \sigma^2) \right). \tag{2.99}$$

[4]We found a small typo in [12], which has been corrected in the above expression.

By combining (2.90) with (2.98), for user 1 we get

$$\begin{aligned}\mu_{vc}^{[1],(\ell+1)} &= \frac{1}{2}\left(F_{+1,+1}(\mu_{vs}^{[2],\ell}) + F_{+1,-1}(\mu_{vs}^{[2],\ell})\right) \\ &\quad + \sum_i \lambda_i^{[1]}(i-1)\left(\sum_j \rho_j^{[1]}\phi^{-1}\left(1-\left(1-\phi(\mu_{vc}^{[1],\ell})\right)^{(j-1)}\right)\right).\end{aligned} \tag{2.100}$$

Similarly, by combining (2.90) with (2.99), for user 2 we get

$$\begin{aligned}\mu_{vc}^{[2],(\ell+1)} &= \frac{1}{2}\left(F_{+1,+1}(\mu_{vs}^{[1],\ell}) + F_{+1,-1}(\mu_{vs}^{[1],\ell})\right) \\ &\quad + \sum_i \lambda_i^{[2]}(i-1)\left(\sum_j \rho_j^{[2]}\phi^{-1}\left(1-\left(1-\phi(\mu_{vc}^{[2],\ell})\right)^{(j-1)}\right)\right).\end{aligned} \tag{2.101}$$

Unequal Power: In the unequal power case, a new rule needs to be derived for the state-to-variable messages since the update rule has changed significantly in comparison with [12]. In fact, the update rule for the means of the state node is now different for each user, since

$$sv^{[1]}(y, vs^{[2]}) = \log \frac{e^{-\frac{(y-\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{vs^{[2]}} + e^{-\frac{(y-\sqrt{P_1}+\sqrt{P_2})^2}{2}}}{e^{-\frac{(y+\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{vs^{[2]}} + e^{-\frac{(y+\sqrt{P_1}+\sqrt{P_2})^2}{2}}}, \tag{2.102}$$

$$sv^{[2]}(y, vs^{[1]}) = \log \frac{e^{-\frac{(y-\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{vs^{[1]}} + e^{-\frac{(y+\sqrt{P_1}-\sqrt{P_2})^2}{2}}}{e^{-\frac{(y-\sqrt{P_1}+\sqrt{P_2})^2}{2}} e^{vs^{[1]}} + e^{-\frac{(y+\sqrt{P_1}+\sqrt{P_2})^2}{2}}}. \tag{2.103}$$

For the sake of the derivation in Appendix A, we need to decide which user transmits the all-ones BPSK codeword and which user transmits the BPSK codeword of type one-half. We assume, without loss of generality, that user 1 transmits the all-ones BPSK codeword. If this choice were reversed, the derivation would involve slightly different computations, but the final result would be the same.

By proceeding as we did in the equal power case, the mean of the state-to-variable messages towards user 2 at iteration ℓ can be calculated as

$$\mu_{sv|+1,+1}^{[1],\ell} = F_+^{[1]}(\mu_{vs}^{[2],\ell}) \quad \text{and} \quad \mu_{sv|+1,-1}^{[1],\ell} = F_-^{[1]}(\mu_{vs}^{[2],\ell}), \tag{2.104}$$

where $F_+^{[1]}(\mu)$ and $F_-^{[1]}(\mu)$ are defined as

$$F_+^{[1]}(\mu) = \frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty} e^{-z^2}\log\left(\frac{1+e^{\sqrt{4\mu+8P_2}z+\mu+2P_2}}{1+e^{-\sqrt{4\mu+8P_2}z-\mu-2P_2-4\sqrt{P_1}\sqrt{P_2}}}\right)dz - \mu + 2(P_1-P_2) \tag{2.105}$$

$$F_-^{[1]}(\mu) = \frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty} e^{-z^2}\log\left(\frac{1+e^{-\sqrt{4\mu+8P_2}z-\mu-2P_2}}{1+e^{\sqrt{4\mu+8P_2}z+\mu+2P_2-4\sqrt{P_1}\sqrt{P_2})}}\right)dz + \mu + 2\left(\sqrt{P_1}-\sqrt{P_2}\right)^2 \tag{2.106}$$

Similarly, the mean of the state-to-variable messages towards user 1 at iteration ℓ can be calculated as

$$\mu_{sv|+1,+1}^{[2],\ell} = F_+^{[2]}(\mu_{vs}^{[1],\ell}) \quad \text{and} \quad \mu_{sv|-1,+1}^{[2],\ell} = F_-^{[2]}(\mu_{vs}^{[1],\ell}), \tag{2.107}$$

where $F_+^{[2]}(\mu)$ and $F_-^{[2]}(\mu)$ are defined as

$$F_+^{[2]}(\mu) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log \left(\frac{1 + e^{\sqrt{4\mu+8P_1}z+\mu+2P_1}}{1 + e^{-\sqrt{4\mu+8P_1}z-\mu-2P_1-4\sqrt{P_1}\sqrt{P_2}}} \right) dz - \mu + 2(P_2 - P_1) \quad (2.108)$$

$$F_-^{[2]}(\mu) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log \left(\frac{1 + e^{\sqrt{4\mu+8P_1}z+\mu+2P_1-4\sqrt{P_1}\sqrt{P_2}}}{1 + e^{-\sqrt{4\mu+8P_1}z-\mu-2P_1}} \right) dz - \mu - 2\left(\sqrt{P_2} - \sqrt{P_1}\right)^2 \quad (2.109)$$

For the sake of readability and text continuity, the exact derivation of the above expressions has been stashed away in Appendix A. Using the law of iterated expectations as previously, the overall mean for user 1 can be calculated as

$$\mu_{sv}^{[1],\ell} = \frac{1}{2}\left(F_+^{[1]}(\mu_{vs}^{[2],\ell}) + F_-^{[1]}(\mu_{vs}^{[2],\ell})\right). \quad (2.110)$$

Accordingly, for user 2, we have

$$\mu_{sv}^{[2],\ell} = \frac{1}{2}\left(F_+^{[2]}(\mu_{vs}^{[1],\ell}) + F_-^{[2]}(\mu_{vs}^{[1],\ell})\right). \quad (2.111)$$

By combining (2.90) with (2.110), for user 1 we get

$$\mu_{vc}^{[1],(\ell+1)} = \frac{1}{2}\left(F_+^{[1]}(\mu_{vs}^{[2],\ell}) + F_-^{[1]}(\mu_{vs}^{[2],\ell})\right) + \sum_i \lambda_i^{[1]}(i-1)\left(\sum_j \rho_j^{[1]} \phi^{-1}\left(1 - \left(1 - \phi(\mu_{vc}^{[1],\ell})\right)^{(j-1)}\right)\right). \quad (2.112)$$

Similarly, by combining (2.90) with (2.111), for user 2 we get

$$\mu_{vc}^{[2],(\ell+1)} = \frac{1}{2}\left(F_+^{[2]}(\mu_{vs}^{[1],\ell}) + F_-^{[2]}(\mu_{vs}^{[1],\ell})\right) + \sum_i \lambda_i^{[2]}(i-1)\left(\sum_j \rho_j^{[2]} \phi^{-1}\left(1 - \left(1 - \phi(\mu_{vc}^{[2],\ell})\right)^{(j-1)}\right)\right). \quad (2.113)$$

2.3.2 EXtrinsic Information Transfer (EXIT) Charts

EXtrinsic Information Transfer (EXIT) charts [17] can be used to accurately analyze the behavior of iterative decoding systems. More specifically, the decoder of an LDPC code consists of elements of two types, namely, variable and function (i.e. check and state) nodes. The evolution of some one-dimensional output metric at a particular type of node can be tracked as a function of the same metric at the input of these nodes. EXIT chart analysis is exact for the BEC where messages are inherently one dimensional. For all other channels, we need to reduce the dimensionality of the problem. In our case, through the Gaussian Approximation, we have reduced the dimensionality of the problem from infinite (whole densities, i.e. continuous functions) to one (the mean).

Mutual information between the messages and the codeword bits we wish to decode has been shown to be a metric capable of very accurately predicting the bevahior of an LDPC decoder [19, 22]. Using this metric, we track the mutual information between the variable-to-check messages and the codeword bits as a function of the mutual information between the check-to-variable and the state-to-variable messages, from which the variable-to-check messages were computed, and the codeword bits. For any value of the mutual information between the input messages and the codeword bits, we want the mutual information between the corresponding output messages and the codeword bits to be larger than this value. This provides a sufficient condition for the decoding to converge to a state of absolute certainty, i.e. a state where the mutual information between the output messages and the codeword bits is equal to one. Absolute certainty is equivalent to zero probability of error.

Consider a variable node of degree i. Recall that, under the GA, all outgoing messages from this variable node, denoted m_{vc}, follow a symmetric Gaussian distribution with mean $\mu = \mu_{sv} + (i-1)\mu_{cv}$ and, due to the Tree Assumption, variance $\sigma^2 = 2\mu = \sigma_{sv}^2 + (i-1)\sigma_{cv}^2$. The mutual information between the random variable X_k corresponding to the BPSK symbol x_k and m_{vc} is (for simplicity, we write x for x_k, X for X_k, m for m_{vc}, and M for the random variable representing m) [18]

$$\begin{aligned} I(X;M) &= H(X) - H(X|M) && (2.114)\\ &= 1 - E\left[\log_2\left(\frac{1}{p_{X|M}(x|m)}\right)\right] && (2.115)\\ &= 1 - \int_{-\infty}^{+\infty} p_{M|X}(m|{+1})\log_2\left(1+e^{-m}\right)dm. && (2.116)\end{aligned}$$

A proof, which only requires the message densities to be symmetric, can be found in Appendix B. When conditioned on $X = +1$, M is distributed according to $\mathcal{N}(\sigma^2/2, \sigma^2)$. This leads to [18, 19]

$$I(X;M) = J(\sigma) \triangleq 1 - \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{+\infty} e^{-\frac{(m-\sigma^2/2)^2}{2\sigma^2}}\log_2(1+e^{-m})dm. \tag{2.117}$$

Good approximations for $J(\sigma)$ and its inverse, denoted $J^{-1}(I)$, can be found in Appendix D. Finally, by applying the GA to the check-to-variable and state-to-variable messages, we get [18, 20]

$$\begin{aligned} I_{EVC}^i(I_{AV}, I_{ES}) &= J\left(\sqrt{\sigma_{sv}^2 + (i-1)\sigma_{cv}^2}\right) && (2.118)\\ &= J\left(\sqrt{(i-1)J^{-1}(I_{AV})^2 + J^{-1}(I_{ES})^2}\right), && (2.119)\end{aligned}$$

where I_{AV} is the mutual information between the check-to-variable messages and the codeword bits, I_{ES} is the mutual information between the state-to-variable messages and the codeword bits, and $I_{EVC}^i(I_{AV}, I_{ES})$ is usually called an *elementary* EXIT function. By averaging all elementary EXIT functions over $\lambda(x)$, we get the overall EXIT function for the variable-to-check messages

$$I_{EVC}(I_{AV}, I_{ES}) = \sum_i \lambda_i I_{EVC}^i(I_{AV}, I_{ES}). \tag{2.120}$$

As stated previously, this EXIT function allows us to compute the average mutual information between the variable-to-check messages and the codeword bits for any given value of the average mutual information between the incoming messages, from which the variable-to-check messages are computed, and the codeword bits.

Accordingly, the EXIT function I_{EVS}^i describing the variable-to-state messages emanating from a degree i variable node can be computed as

$$I_{EVS}^i(I_{AV}) = J\left(\sqrt{i}J^{-1}(I_{AV})\right). \tag{2.121}$$

By averaging over $L(x)$, we get the overall EXIT function for the variable-to-state messages

$$I_{EVS}(I_{AV}) = \sum_i L_i I_{EVS}^i(I_{AV}). \tag{2.122}$$

By exploiting the duality between the check nodes and the variable nodes, which holds exactly for the BEC [21] and with good accuracy for the AWGN channel [22], it can be shown that the EXIT function I_{EC}^{i} describing the messages emanating from a degree i check node can be very well approximated as

$$I_{EC}^{i}(I_{AC}) \approx 1 - J\left(\sqrt{(i-1)}J^{-1}(1 - I_{AC})\right), \tag{2.123}$$

where I_{AC} is the mutual information between the variable-to-check messages and the codeword bits. Again, by averaging over $\rho(x)$, we get the overall EXIT function for the check-to-variable messages

$$I_{EC}(I_{AC}) = \sum_{i} \rho_i I_{EC}^{i}(I_{AC}). \tag{2.124}$$

Note that, due to the iterative nature of the BP decoder, we have $I_{AV} = I_{EC}$ and $I_{AC} = I_{EVC}$.

We now need to derive the EXIT function for the state-to-variable messages. We first briefly present the existing results for the equal power case [12] and we will then proceed with the EXIT functions for the unequal power case.

Equal Power: The mutual information between the state-to-variable messages towards user 1 and the codeword bits is the average mutual information over the two types of equiprobable state nodes. Using (2.93) and (2.95), this can be calculated as [12]

$$I_{ES}^{[1]}(I_{EVS}^{[2]}, \sigma^2) = \frac{1}{2}J\left(\sqrt{2F_{+1,+1}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[2]}\right)^2, \sigma^2\right]}\right) + \frac{1}{2}J\left(\sqrt{2F_{+1,-1}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[2]}\right)^2, \sigma^2\right]}\right). \tag{2.125}$$

Accordingly, for user 2, we have

$$I_{ES}^{[2]}(I_{EVS}^{[1]}, \sigma^2) = \frac{1}{2}J\left(\sqrt{2F_{+1,+1}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[1]}\right)^2, \sigma^2\right]}\right) + \frac{1}{2}J\left(\sqrt{2F_{+1,-1}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[1]}\right)^2, \sigma^2\right]}\right). \tag{2.126}$$

We can now combine (2.120), (2.122), and (2.125) to get the overall EXIT function for the variable-to-state messages of user 1

$$\begin{aligned} I_{EVC}^{[1]}(I_{AV}^{[1]}, I_{AV}^{[2]}, \sigma^2) = \sum_{i} \lambda_i^{[1]} J \Bigg[\Bigg(& (j-1)J^{-1}(I_{AV}^{[1]})^2 \\ & + J^{-1}\left[\frac{1}{2}J\left(\sqrt{2F_{+1,+1}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[2]}\right)^2, \sigma^2\right]}\right) + \frac{1}{2}J\left(\sqrt{2F_{+1,-1}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[2]}\right)^2, \sigma^2\right]}\right)\right]^2 \Bigg)^{1/2} \Bigg], \end{aligned} \tag{2.127}$$

where $I_{EVS}^{[2]} = \sum_i L_i^{[2]} J\left(\sqrt{i}J^{-1}(I_{AV}^{[2]})\right)$. Correspondingly, for user 2 we have

$$\begin{aligned} I_{EVC}^{[2]}(I_{AV}^{[2]}, I_{AV}^{[1]}, \sigma^2) = \sum_{i} \lambda_i^{[2]} J \Bigg[\Bigg(& (i-1)J^{-1}(I_{AV}^{[2]})^2 \\ & + J^{-1}\left[\frac{1}{2}J\left(\sqrt{2F_{+1,+1}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[1]}\right)^2, \sigma^2\right]}\right) + \frac{1}{2}J\left(\sqrt{2F_{+1,-1}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[1]}\right)^2, \sigma^2\right]}\right)\right]^2 \Bigg)^{1/2} \Bigg], \end{aligned} \tag{2.128}$$

where $I_{EVS}^{[1]} = \sum_i L_i^{[1]} J\left(\sqrt{i} J^{-1}(I_{AV}^{[1]})\right)$.

The expressions derived above are a generalization of the expressions found in [12]. These expressions hold for any pair $(\lambda^{[1]}(x), \lambda^{[2]}(x))$ and not just for $\lambda^{[1]}(x) = \lambda^{[2]}(x)$. Unfortunately, as we will see below, they are not particularly convenient for efficient optimization, so in the sequel we shall make the same assumptions as the ones found in [12].

Unequal Power: By using the mean update expressions (2.110) and (2.111) and proceeding in accordance with the equal power case, for the state-to-variable messages towards user 1 we have

$$I_{ES}^{[1]}(I_{EVS}^{[2]}, P_1, P_2) = \frac{1}{2} J\left(\sqrt{2F_{+}^{[1]}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[2]}\right)^2, P_1, P_2\right]}\right) + \frac{1}{2} J\left(\sqrt{2F_{-}^{[1]}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[2]}\right)^2, P_1, P_2\right]}\right). \tag{2.129}$$

Correspondingly, for the state-to-variable messages towards user 2 we have

$$I_{ES}^{[2]}(I_{EVS}^{[2]}, P_1, P_2) = \frac{1}{2} J\left(\sqrt{2F_{+}^{[2]}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[2]}\right)^2, P_1, P_2\right]}\right) + \frac{1}{2} J\left(\sqrt{-2F_{-}^{[2]}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[2]}\right)^2, P_1, P_2\right]}\right). \tag{2.130}$$

By substituting $\sigma = \mu^2/2$ in the approximation of $J(\sigma)$, we have implicitly assumed that $\mu \geq 0$. However, we have also assumed that user 2 transmits a codeword of type one-half, so at variable nodes corresponding to -1 symbols the mean of the incoming messages will be negative. We know that the entropy of a Gaussian random variable does not depend on its mean but solely on its variance. For this reason, we negate $F_{-}^{[2]}$ in the above expression to calculate the mutual information between a density with variance $-2\mu \geq 0$ and the codeword bits.

We can now combine (2.120), (2.122) and (2.129) to get the overall EXIT function for the variable-to-check messages of user 1 as follows

$$\begin{aligned} I_{EVC}^{[1]}(I_{AV}^{[1]}, I_{AV}^{[2]}, P_1, P_2) = \sum_i \lambda_i^{[1]} J \Bigg[\Bigg(&(i-1) J^{-1}(I_{AV}^{[1]})^2 \\ &+ J^{-1}\left[\frac{1}{2} J\left(\sqrt{2F_{+}^{[1]}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[2]}\right)^2, P_1, P_2\right]}\right) + \frac{1}{2} J\left(\sqrt{2F_{-}^{[1]}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[2]}\right)^2, P_1, P_2\right]}\right)\right]^2 \Bigg)^{1/2} \Bigg], \end{aligned} \tag{2.131}$$

where $I_{EVS}^{[2]} = \sum_i L_i^{[2]} J\left(\sqrt{i} J^{-1}(I_{AV}^{[2]})\right)$. Correspondingly, by combining (2.120), (2.122) and (2.130), for user 2 we have

$$\begin{aligned} I_{EVC}^{[2]}(I_{AV}^{[2]}, I_{AV}^{[1]}, P_1, P_2) = \sum_i \lambda_i^{[2]} J \Bigg[\Bigg(&(i-1) J^{-1}(I_{AV}^{[2]})^2 \\ &+ J^{-1}\left[\frac{1}{2} J\left(\sqrt{2F_{+}^{[2]}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[1]}\right)^2, P_1, P_2\right]}\right) + \frac{1}{2} J\left(\sqrt{-2F_{-}^{[2]}\left[\frac{1}{2}J^{-1}\left(I_{EVS}^{[1]}\right)^2, P_1, P_2\right]}\right)\right]^2 \Bigg)^{1/2} \Bigg], \end{aligned} \tag{2.132}$$

where $I_{EVS}^{[1]} = \sum_i L_i^{[1]} J\left(\sqrt{i} J^{-1}(I_{AV}^{[1]})\right)$.

If we draw the EXIT function of the variable nodes with its input on the horizontal axis and its output on the vertical axis, and the EXIT function of the check nodes with its input on the vertical axis and its output on the horizontal axis, the resulting plot is called an EXIT chart.

Chapter 3

Results

In this chapter, we describe the way in which EXIT charts can be used for the optimization of LDPC code ensembles for the two-user GMAC. We present optimization results based on the asymptotic analysis provided by EXIT charts, as well as Monte Carlo simulations for the Bit Error Rate (BER) of specific finite length code instances drawn randomly from the optimized code ensembles.

3.1 Optimization Procedure

As we have already mentioned, a sufficient condition for perfect decoding is that the mutual information between the variable node output messages and the codeword bits (specifically, I_{EVC}) is larger than the mutual information between the variable node input messages and the codeword bits (specifically, I_{AV}) [17]. Since $I_{EC} = I_{AV}$, an equivalent condition is that the inverse of I_{EC} lies strictly below I_{EVC} for both users. Furthermore, it has been shown that it is reasonable to constrain $\rho^{[j]}(x)$ to be concentrated [16], i.e.

$$\rho^{[j]}(x) = x^{k_j}, k_j \in \mathbb{N},\ j = 1, 2.$$

Then, $I_{EC}(I_{AC}) = I_{EC}^{k_j}(I_{AC})$ and the inverse of $I_{EC}(I_{AC})$, denoted $I_{AC}(I_{EC})$, can be easily calculated by solving $I_{EC}(I_{AC})$ for I_{AC} as follows

$$I_{AC}(I_{EC}) = 1 - J\left(\frac{1}{\sqrt{k_j - 1}} J^{-1}(1 - I_{AC})\right). \tag{3.1}$$

This simplification is crucial since the optimization problems which we will state in the sequel would otherwise be non-linear with respect to the coefficients of $\lambda^{[j]}(x)$. This assumption also simplifies the stability conditions so that (2.81) and (2.82) become

$$\lambda_2^{[j]} < \frac{\exp\left(1/(2\sigma^2)\right)}{(k_j - 1)},\ j = 1, 2, \quad \text{and} \quad \lambda_2^{[j]} < \frac{\exp\left(P_j/2\right)}{(k_j - 1)},\ j = 1, 2, \tag{3.2}$$

respectively. In the sequel, we describe the optimization procedure for the equal and unequal power cases.

Equal Power: For the equal power (and equal rate) case, we can assume, without loss of generality, that the two users use codes that belong to the same ensemble [12]. This means that

$$\lambda^{[1]}(x) = \lambda^{[2]}(x) \quad \text{and} \quad \rho^{[1]}(x) = \rho^{[2]}(x) = x^k,\ k \in \mathbb{N}, \tag{3.3}$$

which in turn implies that $L^{[1]}(x) = L^{[2]}(x)$. Since both users are described by exactly the same EXIT chart, we only need to track a single EXIT chart. Thus, we will henceforth drop the user index. Recall that the design rate $r(\lambda(x), \rho(x))$ of an LDPC code ensemble is defined as

$$r(\lambda(x), \rho(x)) = 1 - \frac{\sum_i \rho_i/i}{\sum_i \lambda_i/i}. \tag{3.4}$$

So, maximizing the rate for fixed $\rho(x)$ is equivalent to maximizing $\sum_i \lambda_i/i$, which results in a linear objective function. However, the coefficients of $L(x)$ are a non-linear function of the optimization variables, so the optimization problem can definitely not be expressed as a linear program without making an additional assumption. An assumption which eliminates the need for the calculation of $L(x)$ is the following. Assume that if a state node is connected to a degree i variable node of user 1, then it is also connected to a degree i variable node of user 2. This means that for any variable node of degree i, it holds that

$$I_{EVS}(I_{AV}) = J\left(\sqrt{i} J^{-1}(I_{AV})\right). \tag{3.5}$$

This leads to the following simplified version of (2.127)

$$\begin{aligned} I_{EVC}(I_{AV}, \sigma^2) = \sum_i \lambda_i J \Bigg[\Bigg(&(i-1) J^{-1}(I_{AV})^2 \\ &+ J^{-1}\left[\frac{1}{2} J\left(\sqrt{2F_{+1,+1}\left[\frac{i}{2} J^{-1}(I_{AV})^2, \sigma^2\right]} \right) + \frac{1}{2} J\left(\sqrt{2F_{+1,-1}\left[\frac{i}{2} J^{-1}(I_{AV})^2, \sigma^2\right]} \right) \right]^2 \Bigg)^{1/2} \Bigg], \end{aligned} \tag{3.6}$$

which corresponds exactly to the relation found in [12].

For a given target SNR or, equivalently, noise variance σ^2, we wish to maximize the design rate of the code ensemble while ensuring that BP converges to a vanishingly small probability of error. By setting the maximum allowed variable node degree equal to some $v_{\max} \in \mathbb{N}$ with $v_{\max} > 2$, the overall problem can be formulated as follows

$$\begin{aligned} \underset{\lambda_i}{\text{maximize}} \quad & \sum_{i=2}^{v_{\max}} \lambda_i/i \\ \text{subject to} \quad & I_{AC}(I_{EC}) < \sum_{i=2}^{v_{\max}} \lambda_i I_{EVC}^{i}(I_{AV}, \sigma^2), && (3.7) \\ & \lambda_2 < \frac{\exp\left(1/(2\sigma^2)\right)}{(k-1)}, && (3.8) \\ & \sum_{i=2}^{v_{\max}} \lambda_i = 1, && (3.9) \\ & \lambda_i \geq 0, \quad i = 2, \ldots, v_{\max}. && (3.10) \end{aligned}$$

This is a linear program, which can be solved very efficiently. We observed that the solutions of this linear program are usually sparse, i.e. a limited number of coefficients is non-zero. This property is desirable for simple code construction and decoding.

Unequal Power: In the unequal power case, we can no longer assume that both users employ codes from the same ensemble. We now need to jointly optimize two code ensembles and to find another way to circumvent the non-linearity of $L^{[j]}(x),\ j=1,2$, with respect to the optimization variables. This can be achieved as follows. First, we fix the target SNR for each user, i.e. we fix P_1 and P_2. Note that P_1 and P_2 are system parameters, meaning that they are not to be optimized. Then, we fix $\rho^{[1]}(x)$ and $\rho^{[2]}(x)$ as we did previously. Finally, we fix the variable node degree distribution for one user and optimize the variable node degree distribution of the other user. Since the EXIT chart for one user, say user 1, contains only $L^{[2]}(x)$ and not $L^{[1]}(x)$, this optimization can be formulated as a linear program, as previously. So, for each user we solve the following linear program, having fixed the degree distribution of the other user

$$
\begin{aligned}
\underset{\lambda_i^{[j]}}{\text{maximize}} \quad & \sum_{i=2}^{v_{\max}} \lambda_i^{[j]}/i \\
\text{subject to} \quad & I_{AC}^{[j]} < \sum_{i=2}^{v_{\max}} \lambda_i^{[j]} I_{EVC}^{i,[j]}, && (3.11)\\
& \lambda_2^{[j]} < \frac{\exp\left(P_j/2\right)}{(k_j-1)}, && (3.12)\\
& \sum_{i=2}^{v_{\max}} \lambda_i^{[j]} = 1, && (3.13)\\
& \lambda_i^{[j]} \geq 0,\ i=2,\ldots,v_{\max}. && (3.14)
\end{aligned}
$$

This procedure continues in an alternating fashion until some convergence criterion is met. In extensive simulation studies, we have observed that convergence is reached very quickly and that it is insensitive with respect to the starting point, i.e. the initial degree distribution. We can not guarantee that this method does not get stuck at local maxima. However, as can be seen in the next section, it is very effective. This procedure is repeated for several, well chosen, $(\rho^{[1]}(x), \rho^{[2]}(x))$ pairs and the best result, in terms of sum-rate, is kept.

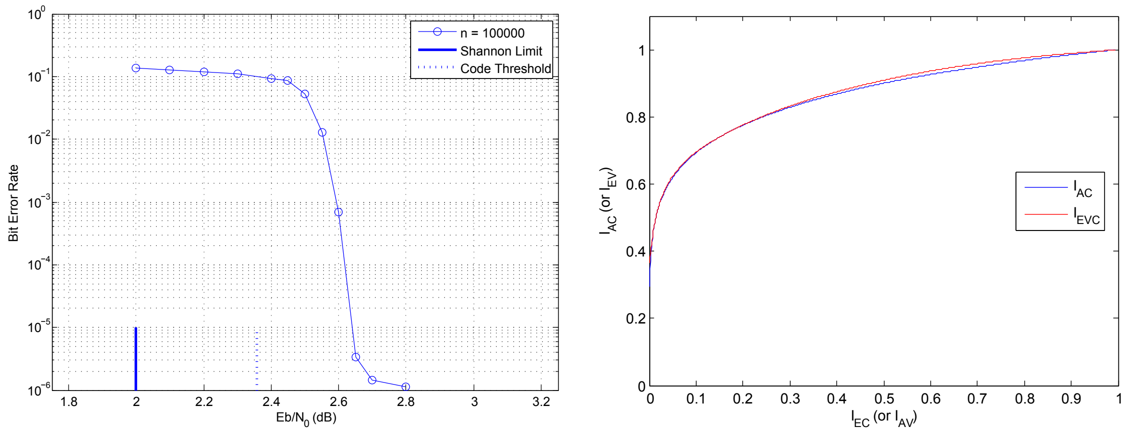
3.2 Results and Finite Length Performance

Equal Power: We set $\sigma = 0.7945$ which corresponds to a sum-rate of 1, i.e. $r = 0.5$ for each user. Optimized degree distributions for the equal power case and for various values of $v_{\max}$ can be seen in Table 3.1. We observe that when we increase $v_{\max}$, the sum-rate increases and for $v_{\max} = 200$ we are only 0.0176 bits per channel use away from the maximal sum-rate.

In order to assess the finite length performance of the optimized ensembles, we created random codes of length $n = 10^5$ for each user with no cycle removal. The maximum number of BP decoding iterations was set to 200. For the code of the user who transmits the codeword of type one-half, we had to create a specific encoder. Thus, a structure which allows for efficient encoding was enforced [4] for this code. Since both codes belong to the same ensemble, their performance is close to the ensemble average with probability that rises exponentially to 1 in the blocklength. So, in Figure 3.1a we only plot the performance of one code for the sake of simplicity. We observe that the codes' performance is about 0.65 dB away from the Shannon limit and about 0.30 dB away from the threshold computed by the EXIT chart method at a BER of 10^{-5}. The error floor is most probably due to the random construction we used.

$\upsilon_{\max}$	50	100	200
λ_2	0.3098	0.2553	0.1841
λ_3	0.3343	0.3138	0.2789
λ_{21}			0.0561
λ_{22}		0.0173	0.1592
λ_{23}		0.2340	
λ_{24}	0.1363		
λ_{25}	0.1144		
λ_{50}	0.1053		
λ_{100}		0.1796	
λ_{200}			0.3216
$\rho(x)$	x^6	x^7	x^8
Rate	0.48734	0.48977	0.49122

Table 3.1: Optimized degree distributions for the equal power case.



(a) BER performance for $n = 10^5$.

(b) EXIT chart.

Figure 3.1: Equal power optimization results for $\upsilon_{\max} = 100$.

Finite-length performance could be further improved, both in the waterfall and in the error floor region, by using a more sophisticated construction method, like the Progressive Edge Growth [23], ACE [24], and ACSE [25] algorithms, or one of their many variants. However, this is beyond the scope of this work. The random construction suffices to demonstrate the potential of this design method, especially for codes with relatively large blocklength.

Unequal Power: For the unequal power case, we explored two scenarios. In the first scenario P_1 and P_2 were relatively close, while, in the second scenario the gap between P_1 and P_2 was significant. Due to the observed insensitivity with respect to the initial distribution, $\lambda^{[1]}(x)$ was always initialized to

$$\lambda^{[1]}(x) = 0.8x + 0.2x^{(\upsilon_{\max}-1)}. \tag{3.15}$$

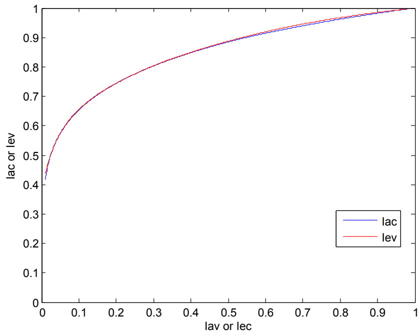
Optimized degree distributions for both scenarios, both users, and for various values of $\upsilon_{\max}$ can be seen in Table 3.2. The corresponding EXIT charts can be seen in subfigures (a) and (b) of Figures 3.2, 3.3, and 3.4.

	$P_1 = 1.5, P_2 = 1$				$P_1 = 1.75, P_2 = 0.75$	
$v_{\max}$	50		200		200	
	$\lambda^{[1]}(x)$	$\lambda^{[2]}(x)$	$\lambda^{[1]}(x)$	$\lambda^{[2]}(x)$	$\lambda^{[1]}(x)$	$\lambda^{[2]}(x)$
λ_2	0.2950	0.3668	0.2429	0.1853	0.1649	0.2159
λ_3	0.3766	0.3529	0.3595	0.2762	0.4754	0.2559
λ_8						0.0320
λ_9						0.1181
λ_{12}		0.0059		0.0489		
λ_{13}		0.1247		0.0705		
λ_{21}	0.3766					
λ_{22}			0.1433		0.2609	
λ_{23}			0.0800			
λ_{32}				0.0569		0.1239
λ_{33}				0.0567		0.0171
λ_{42}		0.1126				
λ_{43}		0.0370				
λ_{50}	0.1408					
λ_{56}					0.0036	
λ_{57}					0.0952	
λ_{98}			0.0631			
λ_{99}			0.1111			
λ_{200}				0.3054		0.2371
	$\rho^{[1]} = x^6$	$\rho^{[1]} = x^4$	$\rho^{[1]} = x^7$	$\rho^{[2]} = x^7$	$\rho^{[1]} = x^{10}$	$\rho^{[2]} = x^5$
Rate	0.4984	0.3644	0.5060	0.3726	0.5756	0.3059

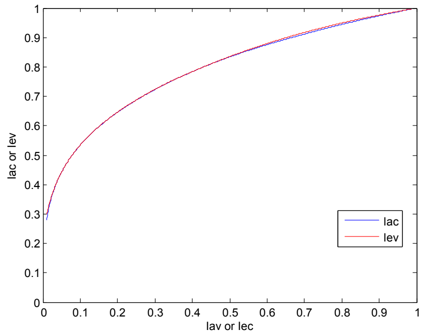
Table 3.2: Optimized degree distributions for the unequal power case.

The maximal sum-rate for the $P_1 = 1.5, P_2 = 1$ scenario is 0.8859 bits/channel use. The code pair ensemble with $v_{\max} = 50$ has a sum-rate which is only 0.0231 bits/channel use away from the maximal sum-rate. By increasing $v_{\max}$ to 200, we can reduce this gap furher to only 0.0073 bits/channel use. In order to assess the first ensemble's finite length performance, we created random codes of length $n = 50000$ for each user. The performance of these codes can be seen in Figure 3.2d. We observe that the resulting BER is less than 10^{-5} at an SNR only 0.55 dB away from the design SNR for both users, even for this moderate blocklength and random construction. In order to assess the second ensemble's finite length performance, we created random codes of length $n = 50000$ for each user. The performance of theses codes can be seen in Figure 3.3d. In this case, performance is even better, since the resulting BER is less than 10^{-5} at an SNR only 0.50 dB away from the design SNR for both users.

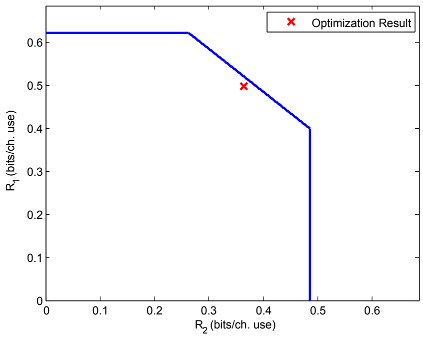
The maximal sum-rate for the scenario where $P_1 = 1.75, P_2 = 0.75$ is 0.8815 bits/channel use. The code pair ensemble with $v_{\max} = 200$ has a sum-rate which is only 0.0019 bits/channel use away from the maximal sum-rate. In order to assess the ensemble's finite length performance, we created random codes of length $n = 50000$ for each user. The performance of theses codes can be seen in Figure 3.4d. In this case, we observe that performance for user 1, who has a higher rate, is similar to the previous cases, i.e. we get a BER of 10^{-5} at an SNR 0.5 dB away from the design SNR. For the low rate user, however, results are not so good, as an error floor starts becoming evident at a BER of 10^{-4}. In our opinion, this can be explained in at least two ways. First, in the low rate region, the Gaussian Approximation has been demonstrated to



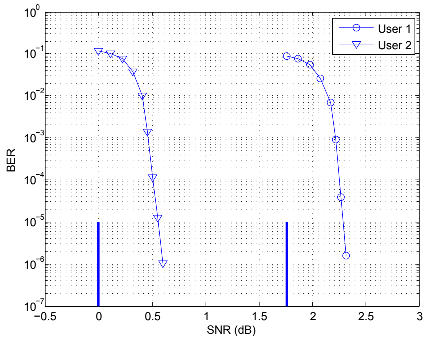
(a) EXIT chart for user 1.



(b) EXIT chart for user 2.



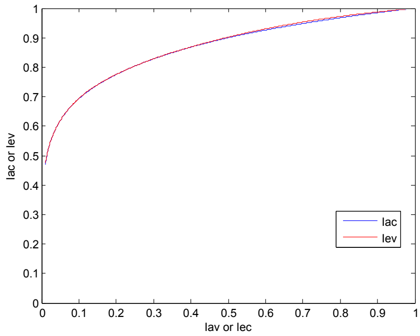
(c) Rate pair and capacity region.



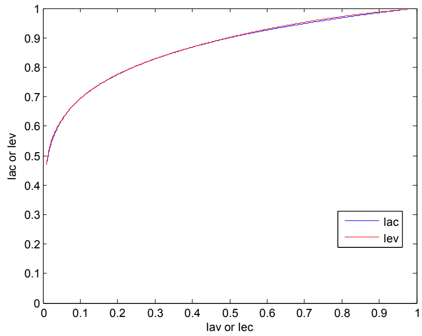
(d) BER performance for $n = 5 \cdot 10^4$.

Figure 3.2: Optimization results for $P_1 = 1.5$ and $P_2 = 1$ and $v_{\max} = 50$.

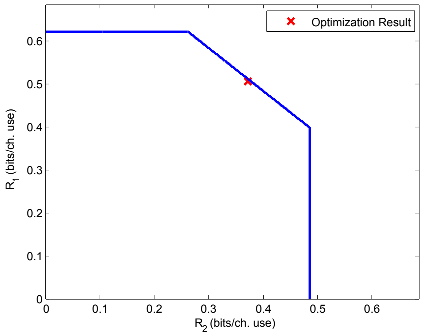
have worse performance than its proposed alternative, the Erasure Channel Approximation [12]. Furthermore, for low rate codes with high $v_{\max}$ random code construction gives poor results, particularly in terms of error floors [26]. This happens mainly because we have a parity-check matrix of larger dimension where a random construction results in significantly more small cycles. This may be the reason why our code has a relatively high error floor. However, performance in the waterfall region is still good and we get a BER of 10^{-5} at approximately 0.9 dB away from the design SNR. Finite-length performance could be further improved by using one of the construction methods mentioned previously.



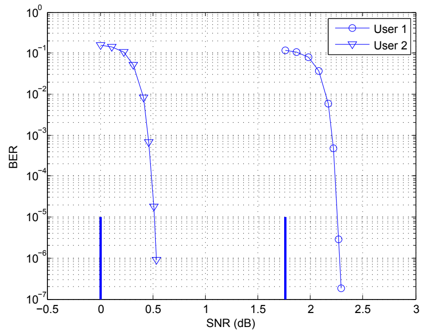
(a) EXIT chart for user 1.



(b) EXIT chart for user 2.

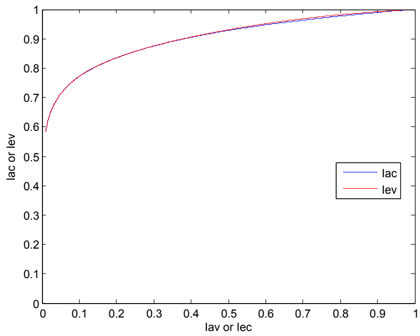


(c) Rate pair and capacity region.

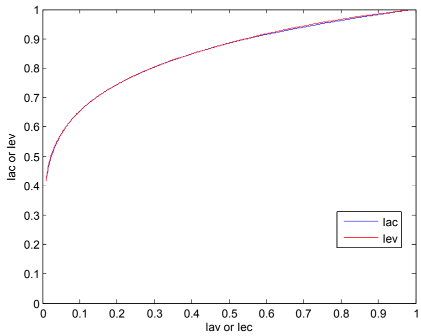


(d) BER performance for $n = 5 \cdot 10^4$.

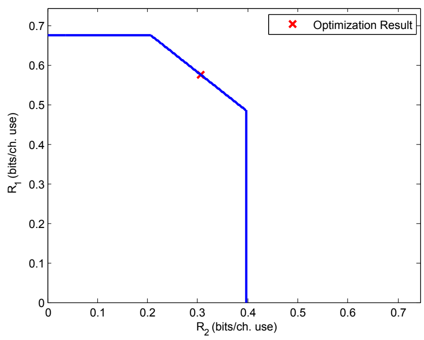
Figure 3.3: Optimization results for $P_1 = 1.5$ and $P_2 = 1$ and $v_{\max} = 200$.



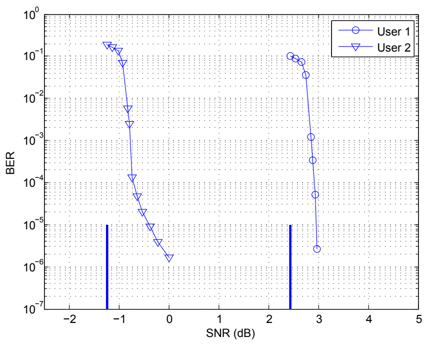
(a) EXIT chart for user 1.



(b) EXIT chart for user 2.



(c) Rate pair and capacity region.



(d) BER performance for $n = 5 \cdot 10^4$.

Figure 3.4: Optimization results for $P_1 = 1.75$ and $P_2 = 0.75$ and $v_{\max} = 200$.

Chapter 4

Conclusion

In this thesis, we developed an EXIT Chart based framework for the optimization of LDPC codes for transmission over the unequal power two-user GMAC channel. We saw that, under some assumptions, the optimization problem can be expressed as an alternating linear programming problem, which can be solved very efficiently and which converges quickly. The resulting codes are close to optimal, in terms of sum-rate, and exhibit very good finite-length behavior.

Future work includes the investigation of the Erasure Channel Approximation [12] for the unequal power GMAC, as well as the derivation and implementation of full Density Evolution for the evaluation of the asymptotic performance of our optimized ensembles. Application of our design framework to larger systems, like relay channels, would also be of interest. A fourth direction is the generalization of our framework to scenarios with more than two users with arbitrary powers. We could also investigate the use of higher order modulations for the GMAC, for example by using Bit-Interleaved Coded Modulation (BICM) [27], in order to achieve rates in the high SNR region.

On a more practical side, a hardware implementation of the joint decoder could also be of interest, mainly due to the complex nature of the state node update rule which would require the use of a good approximation. Ways of constructing good codes of smaller length could also be investigated by modifying existing algorithms (e.g. Progressive Edge Growth [23]) in order to construct small and moderately sized two-user graphs with large girth.

Appendix A

Derivation of $F_+^{[1]}$, $F_-^{[1]}$, $F_+^{[2]}$, and $F_-^{[2]}$

User 1

We will first rewrite the update rule for user 1 in order to proceed with the derivation of the update rule for the means. By dividing both the nominator and the denominator of (2.34) with the second term of the sum in the nominator, we get the following

$$sv^{[1]}(y, vs^{[2]}) = \log \frac{e^{2\sqrt{P_2}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[2]}} + 1}{e^{-2(\sqrt{P_1}-\sqrt{P_2})y + vs^{[2]}} + e^{-(2\sqrt{P_1}\sqrt{P_2} + 2\sqrt{P_1}y)}} \tag{A.1}$$

$$= \log \frac{1}{e^{-2(\sqrt{P_1}-\sqrt{P_2})y + vs^{[2]}}} \frac{e^{2\sqrt{P_2}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[2]}} + 1}{e^{-(2\sqrt{P_2}y + 2\sqrt{P_1}\sqrt{P_2} + vs^{[2]})} + 1} \tag{A.2}$$

$$= 2(\sqrt{P_1} - \sqrt{P_2})y - vs^{[2]} + \log \frac{e^{2\sqrt{P_2}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[2]}} + 1}{e^{-(2\sqrt{P_2}y + 2\sqrt{P_1}\sqrt{P_2} + vs^{[2]})} + 1}. \tag{A.3}$$

We are interested in finding the mean of the above expression, i.e.

$$E\left[sv^{[1]}\right] = E\left[2\left(\sqrt{P_1} - \sqrt{P_2}\right)y - vs^{[2]} + \log \frac{e^{2\sqrt{P_2}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[2]}} + 1}{e^{-(2\sqrt{P_2}y + 2\sqrt{P_1}\sqrt{P_2} + vs^{[2]})} + 1}\right] \tag{A.4}$$

$$= 2E\left[\left(\sqrt{P_1} - \sqrt{P_2}\right)y\right] - E\left[vs^{[2]}\right] + E\left[\log\left(e^{2\sqrt{P_2}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[2]}} + 1\right)\right]$$
$$- E\left[\log\left(e^{-(2\sqrt{P_2}y + 2\sqrt{P_1}\sqrt{P_2} + vs^{[2]})} + 1\right)\right] \tag{A.5}$$

In order to evaluate the mean of the last two terms, we can use the property found in [12] which states that for a Gaussian random variable $w \sim \mathcal{N}(\mu + a, 2\mu + b)$, where a and b are real valued constants, it holds that

$$E[\log(1 + e^{\pm w})] = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log\left(1 + e^{\pm(\sqrt{4\mu + 2b}z + \mu + a)}\right) dz. \tag{A.6}$$

Without loss of generality, we will henceforth assume that User 1 transmits the all-one codeword and User 2 transmits a codeword of type one-half.

For $x^{[1]} = +\sqrt{P_1}$ and $x^{[2]} = +\sqrt{P_2}$, we have

$$y \sim \mathcal{N}\left(\sqrt{P_1} + \sqrt{P_2}, 1\right) \quad \text{and} \quad vs^{[2]} \sim \mathcal{N}(\mu, 2\mu). \tag{A.7}$$

Since $vs^{[2]}$ and y are statistically independent, the distribution of the exponent w of the first term is

$$w \sim \mathcal{N}(2P_2 + \mu, 4P_2 + 2\mu). \tag{A.8}$$

By applying the plus sign version of (A.6) we get

$$E\left[\log\left(e^{(\sqrt{P_1}+\sqrt{P_2})y-2\sqrt{P_1}\sqrt{P_2}+vs^{[2]}}+1\right)\right]=\frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty}e^{-z^2}\log\left(1+e^{\sqrt{4\mu+8P_2}z+\mu+2P_2}\right)dz. \tag{A.9}$$

Accordingly, the distribution of the exponent w of the second term is

$$w\sim\mathcal{N}\left(2P_2+4\sqrt{P_1}\sqrt{P_2}+\mu,4P_2+2\mu\right). \tag{A.10}$$

By applying the minus sign version of (A.6) we get

$$E\left[\log\left(e^{-(2\sqrt{P_2}y+vs^{[2]}+2\sqrt{P_1}\sqrt{P_2})}+1\right)\right]=\frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty}e^{-z^2}\log\left(1+e^{-\sqrt{4\mu+8P_2}z-\mu-2P_2-4\sqrt{P_1}\sqrt{P_2}}\right)dz. \tag{A.11}$$

So, for user 1 the mean update rule for the $(+\sqrt{P_1},+\sqrt{P_2})$ state nodes is the following

$$F_{+}^{[1]}(\mu)=\frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty}e^{-z^2}\log\left(\frac{1+e^{\sqrt{4\mu+8P_2}z+\mu+2P_2}}{1+e^{-\sqrt{4\mu+8P_2}z-\mu-2P_2-4\sqrt{P_1}\sqrt{P_2}}}\right)dz-\mu+2(P_1-P_2).$$

For $x^{[1]}=+\sqrt{P_1}$ and $x^{[2]}=-\sqrt{P_2}$, we have

$$y\sim\mathcal{N}\left(\sqrt{P_1}-\sqrt{P_2},1\right)\quad\text{and}\quad vs^{[2]}\sim\mathcal{N}\left(-\mu,2\mu\right). \tag{A.12}$$

Since $vs^{[2]}$ and y are statistically independent, the distribution of the negation of the exponent w of the first term is

$$-w\sim\mathcal{N}\left(2P_2+\mu,4P_2+2\mu\right). \tag{A.13}$$

By applying the minus sign version of (A.6) we get

$$E\left[\log\left(e^{2\sqrt{P_2}y-2\sqrt{P_1}\sqrt{P_2}+vs^{[2]}}+1\right)\right]=\frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty}e^{-z^2}\log\left(1+e^{-\sqrt{4\mu+8P_2}z-\mu-2P_2}\right)dz. \tag{A.14}$$

Accordingly, the distribution of the exponent w of the second term is

$$w\sim\mathcal{N}\left(2P_2-4\sqrt{P_1}\sqrt{P_2}+\mu,4P_2+2\mu\right). \tag{A.15}$$

By applying the plus sign version of (A.6) we get

$$E\left[\log\left(e^{-(2\sqrt{P_2}y+vs^{[2]}+2\sqrt{P_1}\sqrt{P_2})}+1\right)\right]=\frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty}e^{-z^2}\log\left(1+e^{\sqrt{4\mu+8P_2}z+\mu+2P_2-4\sqrt{P_1}\sqrt{P_2})}\right)dz. \tag{A.16}$$

So, for user 1 the mean update rule for the $(+\sqrt{P_1},-\sqrt{P_2})$ state nodes is the following

$$F_{-}^{[1]}(\mu)=\frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty}e^{-z^2}\log\left(\frac{1+e^{-\sqrt{4\mu+8P_2}z-\mu-2P_2}}{1+e^{\sqrt{4\mu+8P_2}z+\mu+2P_2-4\sqrt{P_1}\sqrt{P_2})}}\right)dz+\mu+2\left(\sqrt{P_1}-\sqrt{P_2}\right)^2. \tag{A.17}$$

User 2

By following the same steps as previously, we can rewrite the update rule for user 2 as follows

$$sv^{[2]}(y, vs^{[1]}) = \log \frac{e^{2\sqrt{P_1}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]}} + 1}{e^{-2(\sqrt{P_2}-\sqrt{P_1})y + vs^{[1]}} + e^{-(2\sqrt{P_2}y + 2\sqrt{P_1}\sqrt{P_2})}} \tag{A.18}$$

$$= \log \frac{1}{e^{-2(\sqrt{P_2}-\sqrt{P_1})y + vs^{[1]}}} \frac{e^{2\sqrt{P_1}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]}} + 1}{e^{-(2\sqrt{P_1}y + 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]})} + 1} \tag{A.19}$$

$$= 2\left(\sqrt{P_2} - \sqrt{P_1}\right) y - vs^{[1]} + \log \frac{e^{2\sqrt{P_1}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]}} + 1}{e^{-(2\sqrt{P_1}y + 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]})} + 1} \tag{A.20}$$

We are interested in finding the mean of the above expression, i.e.

$$E\left[sv^{[2]}\right] = E\left[2\left(\sqrt{P_2} - \sqrt{P_1}\right) y - vs^{[1]} + \log \frac{e^{2\sqrt{P_1}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]}} + 1}{e^{-(2\sqrt{P_1}y + 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]})} + 1}\right] \tag{A.21}$$

$$\begin{aligned} &= 2E\left[\left(\sqrt{P_2} - \sqrt{P_1}\right) y\right] - E\left[vs^{[1]}\right] + E\left[\log\left(e^{2\sqrt{P_1}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]}} + 1\right)\right] \\ &- E\left[\log\left(e^{-(2\sqrt{P_1}y + 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]})} + 1\right)\right] \end{aligned} \tag{A.22}$$

As previously, for $x^{[1]} = +\sqrt{P_1}$ and $x^{[2]} = +\sqrt{P_2}$, we have

$$y \sim \mathcal{N}\left(\sqrt{P_2} + \sqrt{P_1}, 1\right) \quad \text{and} \quad vs^{[1]} \sim \mathcal{N}(\mu, 2\mu). \tag{A.23}$$

Since $vs^{[1]}$ and y are statistically independent, the distribution of the exponent w of the first term is

$$w \sim \mathcal{N}(2P_1 + \mu, 4P_1 + 2\mu). \tag{A.24}$$

By applying (A.6) we get

$$E\left[\log\left(e^{2\sqrt{P_2}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]}} + 1\right)\right] = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log\left(1 + e^{\sqrt{4\mu + 8P_1}z + \mu + 2P_1}\right) dz. \tag{A.25}$$

Accordingly, the distribution of the negation of the exponent w of the second term is

$$-w \sim \mathcal{N}\left(2P_1 + 4\sqrt{P_1}\sqrt{P_2} + \mu, 4P_1 + 2\mu\right). \tag{A.26}$$

By applying the minus sign version of (A.6) we get

$$E\left[\log\left(e^{-(2\sqrt{P_1}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]})} + 1\right)\right] = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log\left(1 + e^{-\sqrt{4\mu + 8P_1}z - \mu - 2P_1 - 4\sqrt{P_1}\sqrt{P_2}}\right) dz. \tag{A.27}$$

So, for user 2 the mean update rule for the $(+\sqrt{P_1}, +\sqrt{P_2})$ state nodes is the following

$$F_{+}^{[2]}(\mu) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log\left(\frac{1 + e^{\sqrt{4\mu + 8P_1}z + \mu + 2P_1}}{1 + e^{-\sqrt{4\mu + 8P_1}z - \mu - 2P_1 - 4\sqrt{P_1}\sqrt{P_2}}}\right) dz - \mu + 2(P_2 - P_1). \tag{A.28}$$

For $x^{[1]} = +\sqrt{P_1}$ and $x^{[2]} = -\sqrt{P_2}$, we have

$$y \sim \mathcal{N}\left(\sqrt{P_1} - \sqrt{P_2}, 1\right) \quad \text{and} \quad vs^{[1]} \sim \mathcal{N}(\mu, 2\mu). \tag{A.29}$$

Since $vs^{[1]}$ and y are statistically independent, the distribution of the exponent w of the first term is

$$w \sim \mathcal{N}\left(2P_1 - 4\sqrt{P_1}\sqrt{P_2} + \mu, 4P_1 + 2\mu\right). \tag{A.30}$$

By applying the plus sign version of (A.6) we get

$$E\left[\log\left(e^{2\sqrt{P_1}y - 2\sqrt{P_1}\sqrt{P_2} + vs^{[1]}} + 1\right)\right] = \frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty} e^{-z^2}\log\left(1 + e^{\sqrt{4\mu+8P_1}z + \mu + 2P_1 - 4\sqrt{P_1}\sqrt{P_2}}\right) dz. \tag{A.31}$$

Accordingly, the distribution of the exponent w of the negation of the second term is

$$-w \sim \mathcal{N}\left(2P_1 + \mu, 4P_1 + 2\mu\right). \tag{A.32}$$

By applying the minus sign version of (A.6) we get

$$E\left[\log\left(e^{-(2\sqrt{P_1}y + vs^{[2]} + 2\sqrt{P_1}\sqrt{P_2})} + 1\right)\right] = \frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty} e^{-z^2}\log\left(1 + e^{-\sqrt{4\mu+8P_1}z - \mu - 2P_1}\right) dz. \tag{A.33}$$

So, for user 2 the mean update rule for the $(+\sqrt{P_1}, -\sqrt{P_2})$ state nodes is the following

$$F_{-}^{[2]}(\mu) = \frac{1}{\sqrt{\pi}}\int_{-\infty}^{+\infty} e^{-z^2}\log\left(\frac{1 + e^{\sqrt{4\mu+8P_1}z + \mu + 2P_1 - 4\sqrt{P_1}\sqrt{P_2}}}{1 + e^{-\sqrt{4\mu+8P_1}z - \mu - 2P_1}}\right) dz - \mu - 2\left(\sqrt{P_2} - \sqrt{P_1}\right)^2. \tag{A.34}$$

Appendix B

Derivation of $I(X;M)$

$$
\begin{aligned}
I(X;M) &= H(X) - H(X|M) && \text{(B.1)}\\
&= 1 - E\left[\log_2\left(\frac{1}{p_{X|M}(x|m)}\right)\right] && \text{(B.2)}\\
&= 1 - \sum_{x=\pm1}\int_{-\infty}^{+\infty} p_{X,M}(x,m)\log_2\left(\frac{1}{p_{X|M}(x|m)}\right)dm && \text{(B.3)}\\
&= 1 - \sum_{x=\pm1} p_X(x)\int_{-\infty}^{+\infty} p_{M|X}(m|x)\log_2\left(\frac{p_M(m)}{p_{M|X}(m|x)P_X(x)}\right)dm && \text{(B.4)}\\
&= 1 - \sum_{x=\pm1} p_X(x)\int_{-\infty}^{+\infty} p_{M|X}(m|x)\log_2\left(\frac{\frac{1}{2}p_{M|X}(m|{+1}) + \frac{1}{2}p_{M|X}(m|{-1})}{p_{M|X}(m|x)p_X(x)}\right)dm && \text{(B.5)}\\
&= 1 - \left[\frac{1}{2}\int_{-\infty}^{+\infty} p_{M|X}(m|{+1})\log_2\left(\frac{\frac{1}{2}p_{M|X}(m|{+1}) + \frac{1}{2}p_{M|X}(m|{-1})}{\frac{1}{2}p_{M|X}(m|{+1})}\right)dm\right.\\
&\left.+\frac{1}{2}\int_{-\infty}^{+\infty} p_{M|X}(m|{-1})\log_2\left(\frac{\frac{1}{2}p_{M|X}(m|{+1}) + \frac{1}{2}p_{M|X}(m|{-1})}{\frac{1}{2}p_{M|X}(m|{-1})}\right)dm\right] && \text{(B.6)}\\
&= 1 - \left[\frac{1}{2}\int_{-\infty}^{+\infty} p_{M|X}(m|{+1})\log_2\left(\frac{p_{M|X}(m|{+1}) + p_{M|X}(m|{-1})}{p_{M|X}(m|{+1})}\right)dm\right.\\
&\left.+\frac{1}{2}\int_{-\infty}^{+\infty} p_{M|X}(-m|{+1})\log_2\left(\frac{p_{M|X}(-m|{-1}) + p_{M|X}(-m|{+1})}{p_{M|X}(-m|{+1})}\right)dm\right] && \text{(B.7)}\\
&= 1 - \int_{-\infty}^{+\infty} p_{M|X}(m|{+1})\log_2\left(1 + \frac{p_{M|X}(m|{-1})}{p_{M|X}(m|{+1})}\right)dm && \text{(B.8)}\\
&= 1 - \int_{-\infty}^{+\infty} p_{M|X}(m|{+1})\log_2\left(1 + \frac{p_{M|X}(-m|{+1})}{p_{M|X}(m|{+1})}\right)dm && \text{(B.9)}\\
&= 1 - \int_{-\infty}^{+\infty} p_{M|X}(m|{+1})\log_2\left(1 + e^{-m}\right)dm && \text{(B.10)}
\end{aligned}
$$

For steps (B.7) and (B.9), we used the fact that symmetry is preserved throughout the decoding process, so the messages m will always have symmetric densities.

Appendix C

Approximation for $\phi(x)$

The following good approximation [16] can be used for $\phi(x)$

$$\phi(x) \approx \begin{cases} e^{\alpha x^{\gamma}+\beta}, & x < 10, \\ \sqrt{\frac{\pi}{x}} e^{-\frac{x}{4}} \left(1 - \frac{20}{7x}\right), & x \geq 10, \end{cases} \tag{C.1}$$

where $\alpha = -0.4527$, $\beta = 0.0218$, and $\gamma = 0.86$. For the inverse function, $\phi^{-1}(\cdot)$, a lookup table can be used.

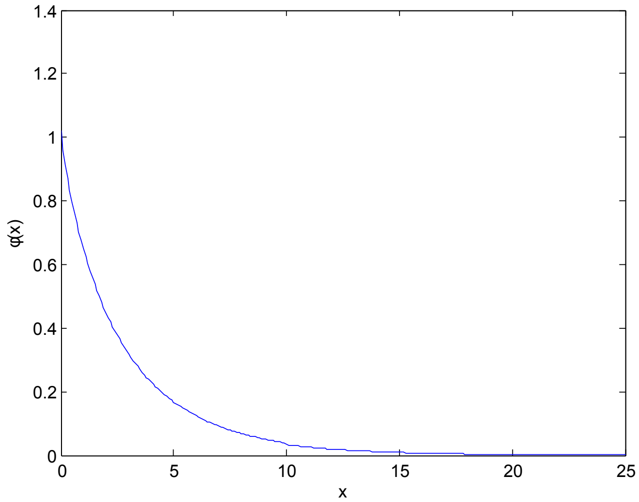


Figure C.1: Approximation for $\phi(x)$.

Appendix D

Approximations for $J(\sigma)$ and $J^{-1}(I)$

The following good approximation [20] can be used for $J(\sigma)$

$$J(\sigma) \approx \begin{cases} a_{j1}\sigma^3 + b_{j1}\sigma^2 + c_{j1}\sigma, & 0 \le \sigma \le \sigma^*, \\ 1 - e^{a_{j2}\sigma^3 + b_{j2}\sigma^2 + c_{j2}\sigma + d_{j2}}, & \sigma^* < \sigma \le 10, \\ 1, & \sigma > 10, \end{cases} \tag{D.1}$$

where $\sigma^* = 1.6363$, $d_{j2} = +0.0549608$, and

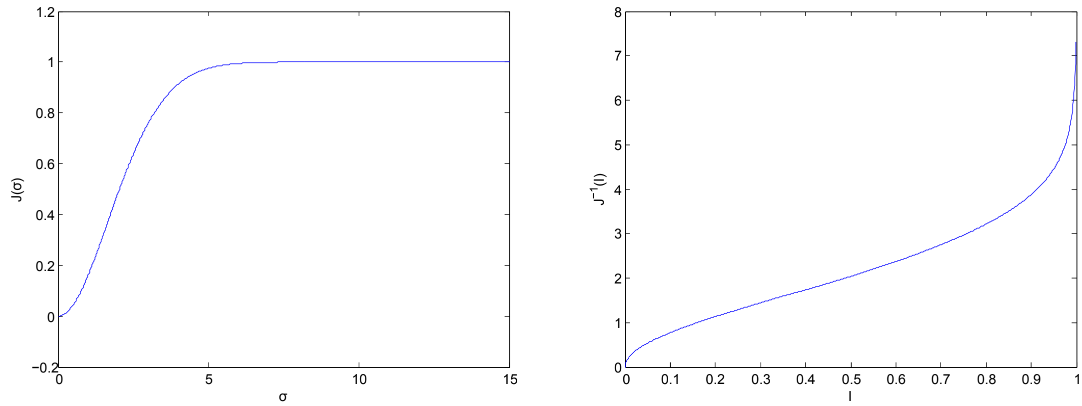
$$\begin{array}{ll} a_{j1} = -0.0421061, & a_{j2} = +0.00181491, \\ b_{j1} = +0.209252, & b_{j2} = -0.142675, \\ c_{j1} = -0.00640081, & c_{j2} = -0.0822054. \end{array} \tag{D.2}$$

The following good approximation [20] can be used for $J^{-1}(I)$

$$J^{-1}(I) \approx \begin{cases} a_{s1}I^2 + b_{s1}I + c_{s1}\sqrt{I}, & 0 \le I \le I^* \\ -a_{s2}\log_2\left(b_{s2}(1-I)\right) - c_{s2}I, & I^* < I \le 1, \end{cases} \tag{D.3}$$

where $I^* = 0.3646$, and

$$\begin{array}{ll} a_{s1} = +1.09542, & a_{s2} = +0.706692, \\ b_{s1} = +0.214217, & b_{s2} = +0.386013, \\ c_{s1} = +2.33727, & c_{s2} = -1.75017. \end{array}$$



(a) $J(\sigma)$ (b) $J^{-1}(I)$

Figure D.1: Approximations for $J(\sigma)$ and $J^{-1}(I)$.

Bibliography

[1] T. M. Cover and A. A. E. Gamal, "Capacity theorems for the relay channel," IEEE Transactions on Information Theory, vol. 25, no. 5, pp. 572-584, Sep. 1979.

[2] R. Gallager, "Low-density parity-check codes," IRE Transactions on Information Theory, pp. 21–28, Jan. 1962.

[3] D. MacKay and R. Neal, "Good codes based on very sparse matrices," in Lecture Notes in Computer Science, pp. 100-111, Cryptography and Coding, 5th IMA Conference, Springer, 1995.

[4] T. Richardson and R. Urbanke, "Efficient encoding of low-density parity-check codes," IEEE Transactions on Information Theory, vol. 47, no. 2, pp. 638-656, Feb. 2001.

[5] M. R. Tanner, "A recursive approach to low complexity codes," IEEE Transactions on Information Theory, vol. 27, pp. 533-547, Sept. 1981.

[6] T. Richardson and R. Urbanke, *Modern Coding Theory*, Cambridge University Press, 2008.

[7] T. Cover and J. Thomas, *Elements of Information Theory*, John Wiley & Sons, Inc., 2006.

[8] A. D. Wyner, "Recent results in the Shannon theory," IEEE Transactions on Information Theory, vol. IT-20, pp. 2–10, Jan. 1974.

[9] P. P. Bergmans and T. M. Cover, "Cooperative broadcasting," IEEE Transactions on Information Theory, vol. IT-20, pp. 317–324, May 1974.

[10] B. Rimoldi and R. Urbanke, "A rate splitting approach to the Gaussian multiple-access channel," IEEE Transactions on Information Theory, vol. 42, no. 2, pp. 364–375, Mar. 1996.

[11] A. Amraoui, S. Dusad, and R. Urbanke, "Achieving general points in the 2-user Gaussian MAC without time-sharing or rate-splitting by means of iterative decoding," in Proceedings of the IEEE International Symposium on Information Theory, 2002, p. 334, July 2002.

[12] A. Roumy and D. Declercq, "Characterization and optimization of LDPC codes for the 2-user Gaussian multiple access channel," EURASIP Journal on Wireless Communications and Networking,, vol. 2007, June 2007.

[13] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*, Morgan Kaufmann, 1988.

[14] T. J. Richardson, M. A. Shokrollahi and R. Urbanke, "Design of capacity approaching irregular low-density parity-check codes," IEEE Transactions on Information Theory, vol. 47, no. 2, pp. 619-637, Feb. 2001.

[15] P. Berlin and D. Tuninetti, "LDPC codes for fading Gaussian broadcast channels," IEEE Transactions on Information Theory, vol. 51, no. 6, pp. 2173–2182, June 2005.

[16] S.-Y. Chung, T. Richardson, and R. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," IEEE Transactions on Information Theory, vol. 47, no. 2, pp. 657–670, Feb. 2001.

[17] M. Tuchler and J. Hagenauer, "EXIT charts of irregular codes," in Proceedings Conference on Information Sciences and Systems, pp. 748–753, Mar. 2002.

[18] G. Liva, S. Song, L. Lan, Y. Zhang, W. Ryan, and S. Lin, "Design of LDPC codes: a survey and new results," *J. Comm. Software and Systems*, Sept. 2006.

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

[20] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," IEEE Transactions on Communications, vol. 52, no. 4, pp. 670-678, Apr. 2004.

[21] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: model and erasure channel properties," IEEE Transactions on Information Theory, vol. 50, no. 11, pp. 2657–2673, Nov. 2004.

[22] E. Sharon, A. Ashikhmin, S. Litsyn, "Analysis of low-density parity-check codes based on EXIT functions," IEEE Transactions on Communications, vol. 54, no. 8, pp. 1407-1413, Aug. 2006.

[23] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge growth Tanner graphs," IEEE Transactions on Information Theory, vol. 51, no. 1, pp. 386-398, Jan. 2005.

[24] T. Tian, C. Jones, J. Villasenor, and R. Wesel, "Construction of irregular LDPC codes with low error floors," in Proceedings IEEE International Conference on Communications, vol. 5, pp. 3125–3129, May 2003.

[25] F. C. M. Xia Zheng Lau, and C. K. Tse, "Constructing short-length irregular LDPC codes with low error floor," IEEE Transactions on Communications, vol. 58, no. 10, pp. 2823–2834, Oct. 2010.

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

[27] G. Caire, G. Taricco, and E. Biglieri, "Bit-interleaved coded modulation," IEEE Transactions on Information Theory, vol. 44, no. 3, pp. 927–946, May 1998.