Fengfan Yang (仰枫帆, 南京航空航天大学) and Tho Le-Ngoc (McGill University)

Chapter 7 General Irregular Low-Density Parity-Check Codes: Fundamentals and Performance Analysis

In this section, we will extend the traditional Gallager's regular LDPC codes to a more general form, i.e., irregular LDPC codes, which have been proved to be able to achieve a better performance approaching to the phenomenal Shannon theoretical limit than those of the optimal regular ones and turbo codes constructed so far by some recent studies [5][6][9]. Similar as the regular LDPC codes, first a mathematical definition of the irregular LDPC codes ensemble and related basic notions are well introduced. Second, the theoretical analysis of iterative decoding for the irregular LDPC codes using Gallager's decoding algorithm A is performed in two simple noisy channels, i.e., pure binary erasure channel (BEC) and binary symmetric channel (BSC), by determination of the erasure probability threshold and bit error probability threshold, respectively. Third, based on the derived threshold we will try to find the optimized irregular LDPC code, which is well characterized by the optimal degree distribution pair for the check and variable nodes of the irregular LDPC codes. Finally, the performance comparison between the regular, irregular LDPC codes and turbo codes are addressed.

7.1 Basic Notions of Irregular LDPC Codes

Recall that an ensemble of regular LDPC codes of length n can be characterized by a pair of fixed parameters (d_v, d_c). The regular LDPC codes are those for which all nodes of the same type (variable or check) have the same degree. For the well-known (3, 6)-

regular LDPC code has a graphical representation in which all variable nodes have degree 3 and all check nodes have degree 6. However, it will be a quite different scenario if we generalize the basic concept of the regular LDPC code to a more general form for construction of an irregular LDPC code, whose degrees chosen based on a pair of degree distributions may vary greatly even if for the same kind of nodes. For example, an irregular LDPC code might have a graphical representation in which half the variable nodes have degree 3 and half have degree 5, while half the check nodes have degree 6 and half have degree 8. Generally, both regular and irregular LDPC codes can be explicitly represented by the well-known bipartite graph, i.e., Tanner graph. The main difference between them only lies in the degree distributions for the variable and check nodes in their Tanner graphs.

In order to well understand the fundamentals of the irregular LDPC codes, some necessary preliminaries, including the important definitions and the illustrative examples, are presented as follows.

7.1.1 Distribution degrees for variable and check nodes:

We define that a polynomial $\gamma(x)$ of the following form

$$\gamma(x)=\sum_{i\geq 2}\gamma_i x^{i-1} \tag{7-1}$$

is a *degree distribution* if $\gamma(x)$ has nonnegative coefficients and $\gamma(1)=1$. Note that we associate the coefficient γ_i to x^{i-1} rather than x^i. We will see in the sequel that this notation, initially introduced by Luby [21], can lead to a very elegant and compact description of the main results of irregular LDPC codes ensemble.

Consider a degree distribution pair $(\lambda(x), \rho(x))$ that associates with the variable and check nodes of an ensemble of LDPC codes $C^n(\lambda, \rho)$ of block length n. The two distribution functions, in terms of polynomials of x, are defined according to the basic polynomial in (7-1) as

$$\lambda(x)=\sum_{i=2}^{d_v}\lambda_i x^{i-1} \tag{7-2a}$$

$$\rho(x)=\sum_{i=2}^{d_c}\rho_i x^{i-1} \tag{7-2b}$$

where the coefficients λ_i and ρ_i represent the fractions of edges emanating from all variable and check nodes of degree i, respectively. Obviously, $\lambda(0)=\rho(0)=0$ and $\lambda(1)=\rho(1)=1$. Alternatively, λ_i (ρ_i) denotes the fraction of ones, which are in columns (rows) of weight i, in the sparse parity-check matrix of an LDPC code.

From the point of view of a variable node, it is best to have high degree, since the more information it gets from its associated check nodes the more accurately it can judge what its correct value should be. Thus, there is no variable node of degree 1 for a good LDPC code, i.e., $\lambda_1=0$.

In contrast, from the point of view of a check node, it is best to have low degree, since the lower degree of a check node, the more valuable the information it can transmit back to its incident variable nodes. However, we do not take the case of a check node of degree 1 into account because the value of the bit involved in the check equation must be 0. This is a trivial case and thus $\rho_1=0$.

The above remarks may intuitively explain the reasons why the indices of both degree distributions in (7-2a) and (7-2b) start from 2. In particular, if a pair of a variable node of degree 1 and a related check node of degree 1 exists in the bipartite graph, this means that only one bit participates in the check equation and the bit is only involved in one check equation. Therefore, we can remove them from the graph without affecting the rest of sub-graph because this pair of incident nodes is completely isolated.

Unlike the regular LDPC codes with parameters d_v and d_c representing the variable and check node degrees, respectively, the polynomials $\lambda(x)$ and $\rho(x)$ for an irregular LDPC code specify the degree distributions of variable and check node, respectively, where, without danger of confusion, d_v and d_c denote the maximum variable and check node degrees, respectively.

For an ensemble of regular LDPC codes $C^{12}(3,6)$, which can be viewed as a special case of the irregular type, we have $\lambda(x)=x^{3-1}$ and $\rho(x)=x^{6-1}$, where λ_3 and ρ_6 are both equal to 1 since all degree for the same type of nodes are the same for the regular LDPC code.

Generally, a regular LDPC code has two degree distributions characterized by polynomials in x for its variable and check nodes, each of polynomials contains only one term associated with constant coefficient 1, while for an irregular LDPC code at least one degree distribution for variable and check node has more than one terms.

7.1.2 Design rate for an irregular LDPC codes ensemble:

Assume E be the total number of edges emanating from all variable nodes of an LDPC code of block length n, the number of variable nodes of degree j is

$$n\frac{(E\lambda_j)/j}{\sum_{i=2}^{d_v}(E\lambda_i)/i}=n\frac{\lambda_j/j}{\sum_{i=2}^{d_v}\lambda_i/i}=n\frac{\lambda_j/j}{\int_0^1\lambda(x)dx} \tag{7-3}$$

where E is equal to

$$E=\sum_{j=2}^{d_v} j\left(n\frac{\lambda_j/j}{\int_0^1\lambda(x)dx}\right)=n\frac{\sum_{i=2}^{d_v}\lambda_i}{\int_0^1\lambda(x)dx}=\frac{n}{\int_0^1\lambda(x)dx} \tag{7-4a}$$

Assume that the LDPC code has m check nodes, E can be similarly expressed by the check node prospect as

$$E=\sum_{j=2}^{d_c} j\left(m\frac{\rho_j/j}{\int_0^1\rho(x)dx}\right)=m\frac{\sum_{j=2}^{d_c}\rho_j}{\int_0^1\rho(x)dx}=\frac{m}{\int_0^1\rho(x)dx} \tag{7-4b}$$

We can conclude the follows if equating the two expressions for E

$$m=n\frac{\int_0^1\rho(x)dx}{\int_0^1\lambda(x)dx} \tag{7-5}$$

Hence, the *design rate* of an LDPC code with block length n and m check equations can be expressed by the degree distributions $\lambda(x)$ and $\rho(x)$ as

$$r(\lambda,\rho)=\frac{n-m}{n}=1-\frac{\int_0^1\rho(x)dx}{\int_0^1\lambda(x)dx} \tag{7-6}$$

Similar as that for the regular LDPC codes, the actual code rate of an irregular LDPC code may be higher than the design rate since the m check equations might not be all linearly independent. Thus, the design rate is the maximum value with all independent check equations.

Example 7.1: For regular LDPC codes ensembles $C^{12}(3,6)$, $C^{12}(4,8)$ and $C^{12}(5,10)$, their design rates are evaluated as follows according to (7-6)

(a) Ensemble of regular LDPC codes $C^{12}(3,6)$

$$r(x^{3-1},x^{6-1})=1-\frac{\int_0^1 x^5dx}{\int_0^1 x^2dx}$$

$$=1-\frac{1/6}{1/3}=\frac{1}{2} \tag{7-7a}$$

(b) Ensemble of Regular LDPC codes $C^{12}(4,8)$

$$r(x^{4-1},x^{8-1})=1-\frac{\int_0^1 x^7dx}{\int_0^1 x^3dx}$$

$$=1-\frac{1/8}{1/4}=\frac{1}{2} \tag{7-7b}$$

(c) Ensemble of Regular LDPC codes $C^{12}(5,10)$

$$r(x^{5-1},x^{10-1})=1-\frac{\int_0^1 x^9 dx}{\int_0^1 x^4 dx}$$

$$=1-\frac{1/10}{1/5}=\frac{1}{2} \tag{7-7c}$$

Apparently, the above results are the same as obtained by (4-2) in Chapter 4. □

Finally, the mathematical definition of an irregular LDPC codes ensemble is given as follows, which admits the regular LDPC codes ensemble defined in previous chapter as a special case.

Definition 7.1: The linear block code is called a binary irregular LDPC code if the degree distributions of the variable and check nodes in its bipartite graph is characterized by the polynomial functions $\lambda(x)$ and $\rho(x)$ given by (7-2a) and (7-2b), where d_v and d_c are denoted as the maximum variable and check node degrees, respectively. The polynomials $\lambda(x)$ and $\rho(x)$ is properly chosen and subject to the constraint $\int_0^1 \lambda(x)dx > \int_0^1 \rho(x)dx$. The number of parity-check equations m is an integer and evaluated by (7-5), where n is the block length of the code. □

Note that in the sequel the individual irregular LDPC code with a pair of degree distribution $(\lambda(x), \rho(x))$ is denoted by $A(n,\lambda,\rho)$ and all these individuals form an ensemble of codes denoted by $C^n(\lambda,\rho)$. For a regular LDPC code, we can simply use the notations $A(n,d_v,d_c)$ and $C^n(d_v,d_c)$ to denote the individual code and codes ensemble, respectively.

Example 7.2: Consider an irregular LDPC code of block length 10, whose degree distributions for the variable and check nodes are $\lambda(x)=\frac{1}{2}x+\frac{1}{2}x^2$ and $\rho(x)=\frac{1}{2}x^2+\frac{1}{2}x^3$, respectively. Hence, the code has a bipartite graph representation in which half the edges emanate from the variable nodes of degree 2 and half emanate from the nodes of degree 3 of the same type, while half the edges emanate from the check nodes of degree 3 and half emanate from the same type nodes of degree 4. According to (7-5) the number of the parity-check equations is

$$m=n\frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx}=10\times\frac{\int_0^1 (x^2+x^3)dx}{\int_0^1 (x+x^2)dx}$$

$$=10\times\frac{1/3+1/4}{1/2+1/3}=7 \tag{7-8a}$$

Obviously, the design rate of the code is

$$r(\lambda,\rho)=1-\frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx}$$

$$=1-\frac{m}{n}=\frac{3}{10} \tag{7-8b}$$

The number of edges in the bipartite graph for the irregular code is computed by (7-4a) from the variable-node prospect as

$$E = \frac{n}{\int_0^1 \lambda(x)dx} = \frac{20}{\int_0^1 \left(x + x^2\right)dx}$$

$$= \frac{20}{1/2 + 1/3} = 24 \qquad \text{(7-9a)}$$

Alternatively, we can get the same result by (7-4b) from the check-node prospect as

$$E = \frac{m}{\int_0^1 \rho(x)dx} = \frac{14}{\int_0^1 \left(x^2 + x^3\right)dx}$$

$$= \frac{14}{1/3 + 1/4} = 24 \qquad \text{(7-9b)}$$

An example of such an irregular LDPC code whose parity-check matrix fulfills the requirements of the degree distributions for variable and check nodes is

$$\mathbf{H}_{7\times10} = \begin{array}{cccccccccc|c} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 & v_8 & v_9 & v_{10} & \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & c_1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & c_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & c_3 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & c_4 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & c_5 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & c_6 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & c_7 \end{array} \qquad \text{(7-10)}$$

where the variables v_i and c_j $(1 \le i \le 10,\ 1 \le j \le 7)$ are clearly marked for each column and row in relation to the variable and check nodes of the associated bipartite graph as shown in Fig. 7.1, respectively.

We can easily obtain from the example that the numbers of variable nodes of degree 2 and 3 are 6 and 4, respectively, while the numbers of check nodes of degree 3 and 4 are 4 and 3, respectively. Hence, the bipartite graph associated with the parity-check matrix in (7-10) is illustrated as follows □

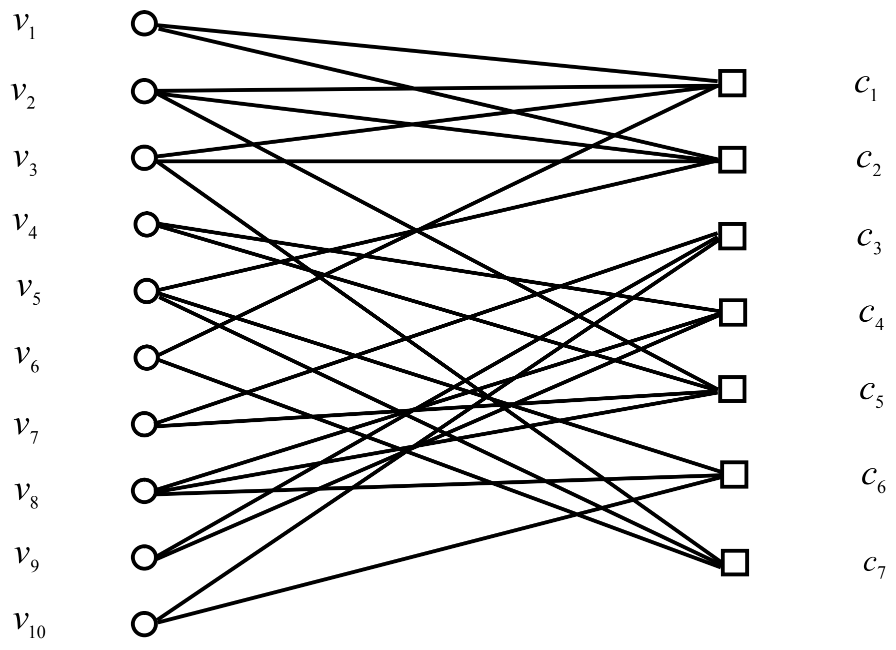


Fig. 7.1 An example of $A(n,\lambda,\rho)$ irregular LDPC code of length n=10 with design rate 3/10 represented by its bipartite graph, where $\lambda(x)=(x+x^2)/2$ and $\rho(x)=(x^2+x^3)/2$. The numbers of variable and check nodes are 10 and 7, respectively.

Note that there could be other variants of irregular LDPC codes whose bipartite graphs also fulfil the degree distributions as above for the variable and check nodes. Therefore, given a pair of degree distribution $(\lambda(x),\rho(x))$, we may find more than one parity-check matrix (or bipartite graph) that all correspond to this distributions pair. Conversely, given a parity-check matrix of an irregular LDPC code, there has an unique pair of ($\lambda(x)$, $\rho(x)$) for itself.

7.2 Analyzing Irregular LDPC Codes from Gallager's Decoding Algorithm A over a Pure BEC

In order to well understand the iterative decoding process for the general class of irregular LDPC codes, we still apply Gallager's decoding algorithm A with hard decision to them over a pure binary erasure channel (BEC), which is the simplest channel model without crossover probability as binary symmetric channel (BSC). We will see that despite the simplicity of the BEC model many of its iterative decoding properties are shared by the general channels considered in this chapter. Together with the case with erasures in the decoder that is investigated in Section 5.3.3, we will present the details of the basic steps in analyzing irregular LDPC codes in taking the varying degrees of variable and check nodes into account.

Let us suppose an irregular LDPC code in relation to a bipartite graph with a degree distribution pair ($\lambda(x)$, $\rho(x)$) for its variable and check nodes, respectively, or alternatively, two real nonnegative sequences (λ_2, λ_3, ..., λ_{d_v}) and (ρ_2, ρ_3, ….., ρ_{d_c}) subject to the constraints $\sum_{i=2}^{d_c}\lambda_i$ =1 and $\sum_{i=2}^{d_v}\rho_i$ =1 with d_v and d_c as the maximum degrees for the variable and check nodes, respectively. The pure BEC model is shown in Fig. 7.2 with binary-input ternary-output and initial erasure probability ε ($0\le\varepsilon\le1$).

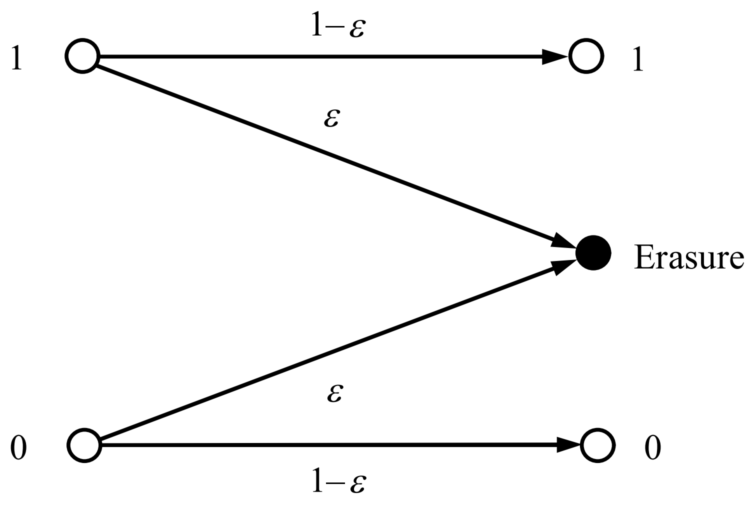


Fig. 7.2 A pure binary-input ternary-output erasure channel (BEC) with an erasure depicted by a black dot symbol.

7.2.1 Gallager's decoding algorithm A over a pure BEC

Gallager's decoding algorithm A for an irregular LDPC code over a pure BEC is quite similar as that for a BSC, which are introduced in Section 5.3. However, we need to take the erasures into account in BEC when the check- and variable-node updates are evaluated in each decoding iteration. First, let us have a brief introduction concerning the essential processing in the two updates, which may differ slightly from the corresponding ones over BSC.

(A) Check-node update: For an arbitrary check node with degree j $(1\le j\le d_c)$ (j incident variable nodes) in an irregular LDPC code, it sends a message to an incident variable node based on the $j-1$ incoming messages related to the other $j-1$ variable nodes. The result from the check node to the associated variable node is the modulo-2 sum of the $j-1$ incoming messages if no erasure is in the incoming messages, which is the same as that of Gallager's decoding A over BSC in this situation. Otherwise, the associated variable node is given an erasure as the result from the check node, which implies that the check node only present an erasure if any of the incoming messages is an erasure. More specifically, it yields

$$\Phi_c\left(m_1, m_2, \ldots\ldots, m_{j-1}\right) = \begin{cases} \sum_{k=1}^{j-1} \oplus m_k & if \quad m_k \neq erasure \;\; for \;\; all \;\; 1 \leq k \leq j-1 \\ erasure & if \qquad\qquad\qquad otherwise \end{cases} \tag{7-11}$$

where $\Phi_c(.)$ is the mapping function of a check node, and $m_k \in \{0,\ 1,\ \text{erasure}\}$ ($1 \leq k \leq j-1$) are the incoming messages given by $j-1$ incident variable nodes in the last decoding iteration. It is quite natural that the result is completely uncertain if one or more erasures exist in the modulo-2 sum. As a consequence, the check node has no choice but to generate an erasure for the incident variable node in this case.

(B) Variable-node update: For an arbitrary variable node with degree i ($1 \leq i \leq d_v$) (i incident check nodes) in an irregular LDPC code, it sends an erasure to an incident check node if and only if the following requirement is fulfilled.

$$\Phi_v\left(m_0, m_1^{'}, m_2^{'}, \ldots\ldots, m_{i-1}^{'}\right) = erasure \tag{7-12a}$$

if $m_0 = m_k^{'} = erasure$ for all $1 \leq k \leq i-1$, where $\Phi_v(.)$ is the mapping function of a variable node, m_0 and $m_k^{'}$ ($1 \leq k \leq i-1$) are the BEC hard-decision output and $i-1$ incoming messages from the other incident check nodes except the one along the same edge as the outgoing message. The decoding strategy in (7-12a) implies that the variable node sends an erasure to the incident check node when the output m_0 related to this variable node and the other $i-1$ incoming messages are all erasures. In other words, the output erasure from the BEC is only confirmed by the fact supported by all the other $i-1$ incident check nodes except the one along the same edge as the outgoing message.

On the contrary, if the output erasure m_0 is not confirmed by all the $i-1$ check nodes, then the incident check node can be given a binary symbol by either 0 or 1 with equal likelihood. This strategy is reasonable because an erasure from the BEC can be decided by a hard decision (without aid from the decoder) as 0 or 1 with equal probability provided that the binary information symbols are equal likelihood in the source. We express this relation as

$$\Phi_v\left(m_0, m_1', m_2', \ldots\ldots, m_{i-1}'\right) = 0 \text{ or } 1 \tag{7-12b}$$

if m_0 is an erasure and at least one m_k' $(1 \le k \le i-1)$ is either 0 or 1.

If m_0 is not an erasure but belongs to GF(2), then the variable node sends its output to the incident check node as

$$\Phi_v\left(m_0, m_1', m_2', \ldots\ldots, m_{i-1}'\right) = \begin{cases} \overline{m}_0 & if \quad m_k' = \overline{m}_0 \quad for \quad all \quad 1 \le k \le i-1 \\ m_0 & if \qquad\qquad otherwise \end{cases} \tag{7-12c}$$

The above expression is quite similar as that of (5-20) for Gallager's decoding algorithm A over BSC. Note that the only slight difference between them is that m_k' may take value as an erasure for some k $(1 \le k \le i-1)$ in BEC.

The decisions by (7-12a), (7-12b) and (7-12c) summarize the output for the variable-node update under various circumstances. In this Section, we are only interested in the erasure probability related to (7-11) and (7-12a), by which the evolution of the erasure probability in the iterative decoding and the related threshold for an irregular LDPC code are derived in theory.

7.2.2 The erasure probability of an LDPC code by Gallager's decoding algorithm A

(i) Regular LDPC codes: For a (d_v, d_c) regular LDPC code in BEC and based on the principle of check-node update, the event, which any check node c_m sends its outgoing message in GF(2) to an incident variable node v_n, is equivalent to that all the variable nodes participating in c_m other than v_n along the same edge as the outgoing message are not erasures. From this point of view, in the first decoding iteration the probability of such an event is simply as $(1-\varepsilon)^{d_c-1}$.

Similarly, based on the principle of variable-node update and in the first decoding iteration, the event of a variable node v_n sending its outgoing erasure to an incident check node c_m is equivalent to that all the check nodes incident to v_n other than c_m along the same edge as the outgoing message are all erasures as well as the output erasure from BEC with respect to v_n. Thus, in the first decoding iteration the probability of such an event is

$$p_{erasure}^{(1)} = \varepsilon\left[1-(1-\varepsilon)^{d_c-1}\right]^{d_v-1} \tag{7-13}$$

Generically, the probability of an outgoing erasure from a variable node to an associated check node in the lth decoding iteration is expressed by a recursive manner as

$$p_{erasure}^{(l)} = \varepsilon\left[1-\left(1-p_{erasure}^{(l-1)}\right)^{d_c-1}\right]^{d_v-1} \tag{7-14}$$

(ii) Irregular LDPC codes

For an irregular LDPC code with a degree distribution pair ($\lambda(x), \rho(x)$), we should slightly modify the above results of the variable-node and check-node updates for a regular LDPC code in order to derive $p_{erasure}^{(l)}$ for an irregular LDPC code.

For a check node of degree j ($1 \le j \le d_c$) in an irregular LDPC code, the probability of an outgoing message taking value in GF(2) at the first iteration is $(1-\varepsilon)^{j-1}$. This gives rise to a general problem to find the probability for an arbitrary check node of all possible degrees. Therefore, we should average the probability $(1-\varepsilon)^{j-1}$ over all the possible degrees for $1 \le j \le d_c$ as.

$$\rho_j \sum_{j=2}^{d_c} (1-\varepsilon)^{j-1} = \rho(x)\big|_{x=1-\varepsilon} = \rho(1-\varepsilon) \tag{7-15}$$

Similarly, based on (7-15) the probability for an arbitrary variable node, which results in an erasure to an incident check node in the first iteration, is also evaluates by averaging over all the possible degrees i ($1 \le i \le d_v - 1$) as

$$p_{erasure}^{(1)} = \varepsilon \sum_{i=2}^{d_v} \lambda_i \left[1-\rho(1-\varepsilon)\right]^{i-1} = \varepsilon\lambda\left(1-\rho(1-\varepsilon)\right) \tag{7-16}$$

Generically, the erasure probability of an outgoing message from a variable node to an incident check node in the lth decoding iteration is recursively evaluated as

$$p_{erasure}^{(l)} = \varepsilon\lambda\left(1-\rho\left(1-p_{erasure}^{(l-1)}\right)\right) \tag{7-17}$$

Let us define a function $f(x,y)$ over [0, 1] as

$$f(x,y)=x\lambda(1-\rho(1-y)) \tag{7-18}$$

It is easily to see that $f(x,y)$ is a strictly increasing function in both its arguments over the entire interval.

7.2.3 The erasure probability threshold for an LDPC code using Gallager's decoding algorithm A over a pure BEC

In this part, we are interested in finding the supremum, denoted by ε^*, of all initial erasure probability ε in [0, 1], such that $p_{erasure}^{(l)}$ in (7-17) converges to zero as l tends to infinity for $0 \le \varepsilon \le \varepsilon^*$. We would ignore the trivial cases for ε =0 and ε =1 since they definitely correspond to $p_{erasure}^{(l)}$=0 and $p_{erasure}^{(l)}$=1 for all l's, respectively. Hence, by (7-18) the following inequalities are valid for $0<\varepsilon<\varepsilon^*<1$ if $p_{erasure}^{(1)} > p_{erasure}^{(0)} = \varepsilon$, i.e.,

$$\varepsilon = p_{erasure}^{(0)} > p_{erasure}^{(1)} > p_{erasure}^{(2)} > \ldots\ldots > p_{erasure}^{(l)} > \ldots\ldots > p_{erasure}^{(\infty)} \tag{7-19}$$

Note that the above inequalities strictly hold for all l's, otherwise if $p_{erasure}^{(m+1)} = p_{erasure}^{(m)}$ for some m, then as a consequence $p_{erasure}^{(l)} = p_{erasure}^{(m)}$>0 for all $l \ge m$. The erasure probability $p_{erasure}^{(l)}$ strictly decreasing with l does not necessarily mean that $p_{erasure}^{(\infty)}$ asymptotically converge to zero. The reader will have a better understanding for it after reading the Example 7.3.

Let us look at an alternative description of the threshold ε_0^* that is closely related to the erasure probability threshold ε^*.

Definition 7.2: The threshold ε_0^* is the supremum of all p_0 in [0, 1] such that

$$x = p_0 \lambda(1 - \rho(1 - x)) \tag{7-20}$$

does not have a strictly positive solution x with $x \le p_0$. □

We can easily prove that the erasure probability threshold ε^* is exactly equal to the threshold ε_0^* as defined by the definition 7.2. The proof for this critical statement is similar as that for the bit error rate (BER) threshold over BSC, which is given by the Theorem 7.3 in the Appendix H.

Based on this statement the erasure probability threshold $\varepsilon^*(\lambda, \rho)$ for an irregular LDPC code in a pure BEC can be alternatively specified by the definition 7.2 instead of finding ε^* directly, which implies that $\varepsilon^*(\lambda, \rho)$ is the minimum value of all p_0 in [0, 1] such that the equation in (7-20) has at least a strictly position solution x with $x \le p_0$. Therefore, p_0 is a function of x, which is denoted by $g(x)$ and expressed as

$$g(x) = \frac{x}{\lambda(1 - \rho(1 - x))} \tag{7-21}$$

Note that $g(1)$ is equal to 1 since $\lambda(1-\rho(1-1)=1$. The limit of $g(x)$ exists as x approaches to zero and $g(x)$ can be defined by using the phenomenal l'Hôpital's rule as,

$$g(0)=\lim_{x\to 0}\frac{x'}{\lambda'(1-\rho(1-x))}=\frac{1}{\lambda_2\rho'(1)} \tag{7-22}$$

Therefore, $g(x)$ is well defined and continuous on [0, 1]. The threshold $\varepsilon^*(\lambda,\rho)$ is thus determined by

$$\varepsilon^*(\lambda,\rho)=\min_{0\le x\le 1}\{g(x): g(x)\ge x\} \tag{7-23}$$

Example 7.3: Consider a (2, 3) regular LDPC codes ensemble over a pure binary erasure channel (BEC), whose degree distributions are defined by $\lambda(x)=x$ and $\rho(x)=x^2$ for variable and check nodes in its bipartite graph, respectively. According to (12.162), the function $g(x)$ is

$$g(x)=\frac{x}{\lambda(1-\rho(1-x))}$$

$$=\frac{x}{1-(1-x)^2}=\frac{1}{2-x} \tag{7-24}$$

Hence, the erasure probability threshold for this regular LDPC code is $\varepsilon^*(2,3)$=1/2 when $g(x)$ achieves its minimum value 1/2 at x=0, and obviously fulfill the constraint $g(x)>x$.

Next, we define a function $h(x)$ for the (2, 3) regular LDPC codes ensemble over the interval [–1, 1]. That is

$$h(x)=c\lambda(1-\rho(1-x))$$

$$=c\left(2x-x^2\right) \tag{7-25}$$

where the coefficient c takes on value in (0, 1) as initial erasure probability and, $h(0)=0$ and $h(1)=c$. Apparently, the function $h(x)$ is a parabola for any $c\neq0$. Fig. 7.3 depicts the function $h(x)$ with c equal to 2/3, 1/2 and 1/3 that represent the cases of initial erasure probability ε greater, equal and less than the corresponding threshold $\varepsilon^*(2,3)=1/2$, respectively. Meanwhile, $h(x)$ with $c=1$ and the straight line $y=x$ are also plotted for the purpose of comparison. Fig. 7.4 (a)~(c) present an intuitive approach to effectively demonstrate the iterative process for each aforementioned value for c as the initial erasure probability ε (or $p_{erasure}^{(0)}$).

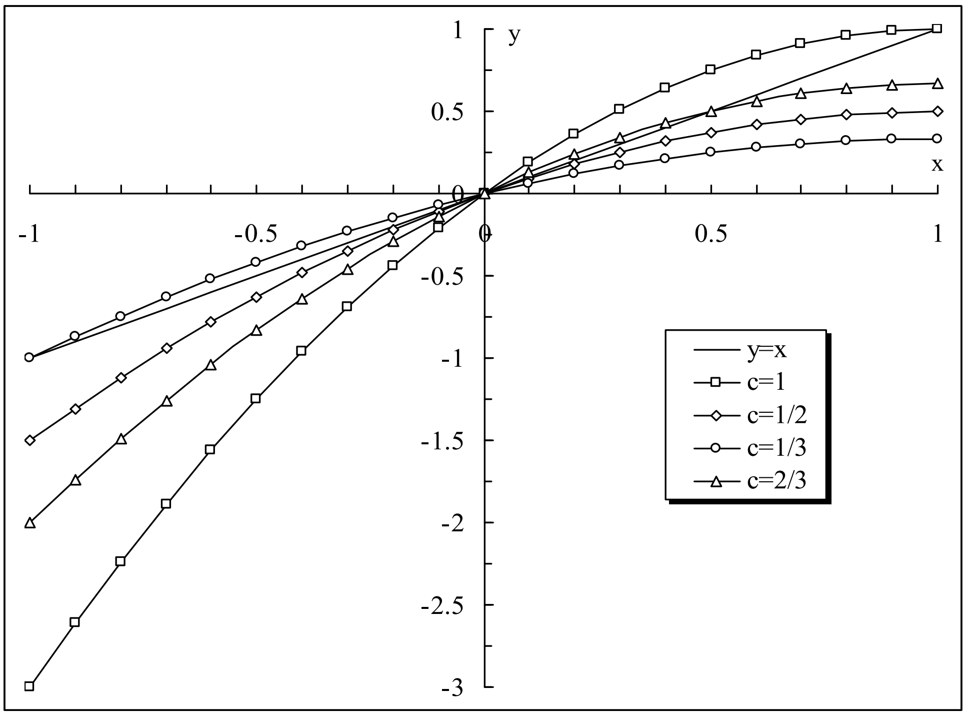


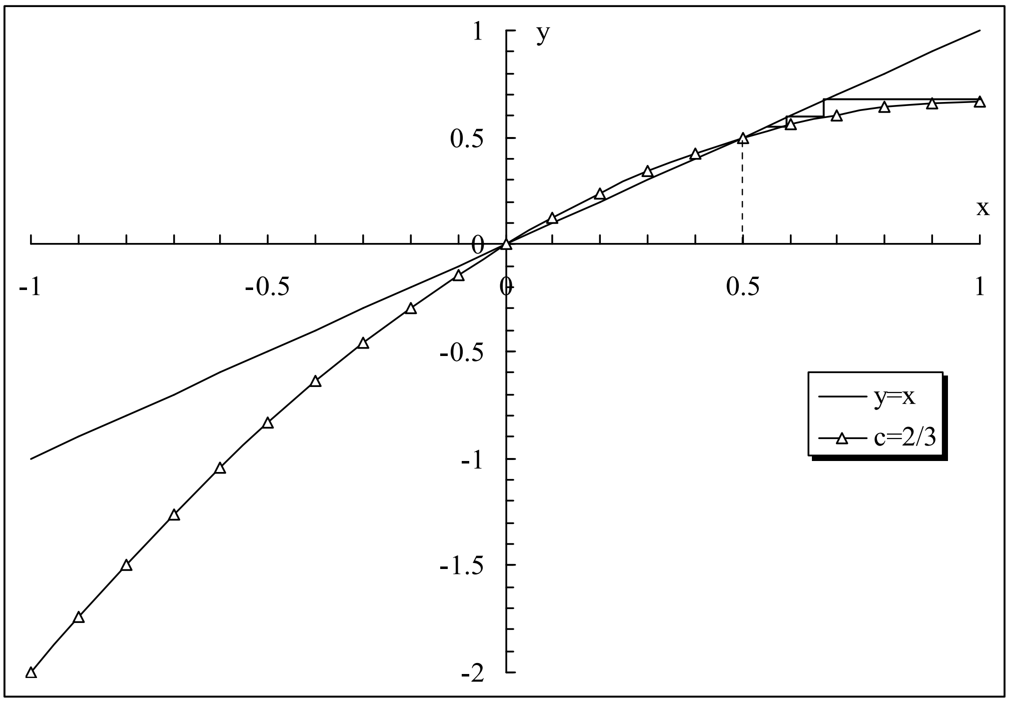
Fig. 7.3 The quadratic function $h(x)$ with c=2/3, 1/2 and 1/3 as various initial erasure probabilities that represent the cases of greater, equal and less than the associated threshold $\varepsilon^*=1/2$, respectively, and $h(x)$ with $c=1$ and the straight line $y=x$ are also plotted for the purpose of comparison.

Fig. 7.4(a) shows that the iterative process with c=2/3 as initial erasure probability, which is greater than the threshold ε^*=1/2, cannot converge to zero erasure probability ($p_{erasure}^{(\infty)}$=0) as the number of iterations trends to infinity, but asymptotically converge to $p_{erasure}^{(\infty)}$=0.5 corresponding to the point at x=0.5, which is intersected by the straight line $y=x$ and indicated by the dashed line as shown in the figure. Note that the abscissa of this intersection point is just the asymptotic erasure probability, i.e., $p_{erasure}^{(\infty)}$=0.5. From this point of view, the erasure probability $p_{erasure}^{(l)}$ monotonically decreasing with the increasing number of iterative decoding does not necessarily guarantee that $p_{erasure}^{(\infty)}$ must converge to zero asymptotically.

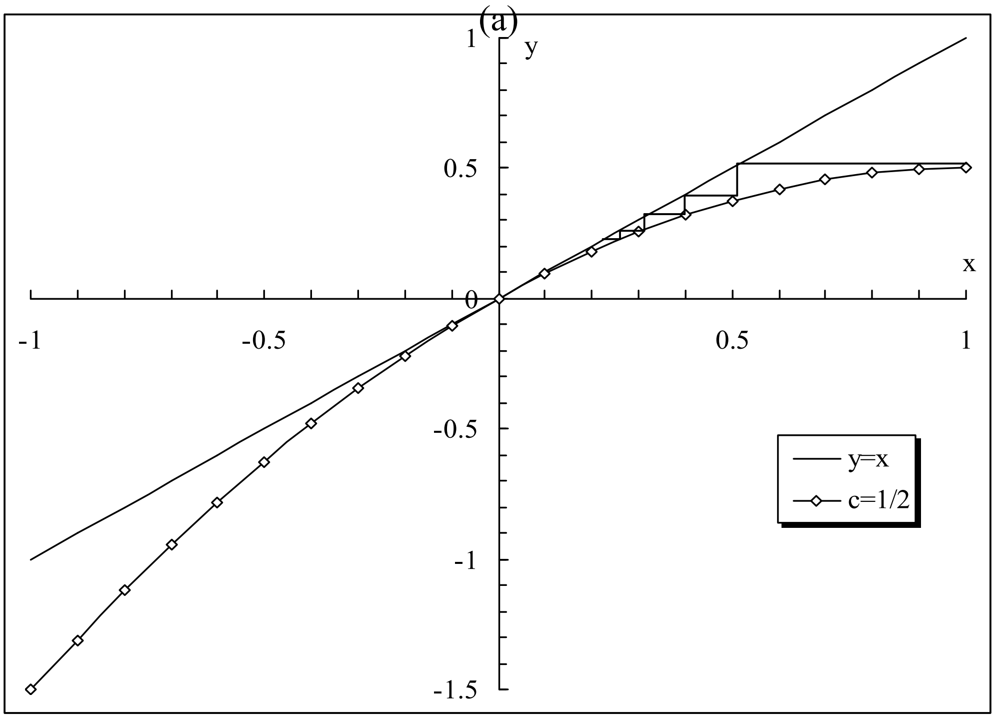
Fig. 7.4(b) presents that the iterative process with c=1/2 as initial erasure probability, which is equal to the threshold ε^*=1/2, can realize zero erasure probability with the infinite number of iterations. Note that the line $y=x$ is tangent to the parabola $h(x)$ with c=1/2 at x=0, which is exactly the convergent point of the asymptotic erasure probability as shown in the figure. This is an interesting case that the asymptotic erasure-free transmission can be achieved with the initial erasure probability equal to the threshold.

Fig. 7.4(c) demonstrates that the iterative process with c =1/3 as initial erasure probability, which is less than the threshold ε^*=1/2, can also reach the goal of erasure-free transmission with the infinite number of iterations. Note that the line $y=x$ intersects the parabola $y=h(x)$ with c=1/3 at x=0 that is the only intersection point in [0, 1], while another point intersected by $y=x$ is at x=–1. The asymptotic erasure probability $p_{erasure}^{(\infty)}$ converges to zero as shown in the figure. □

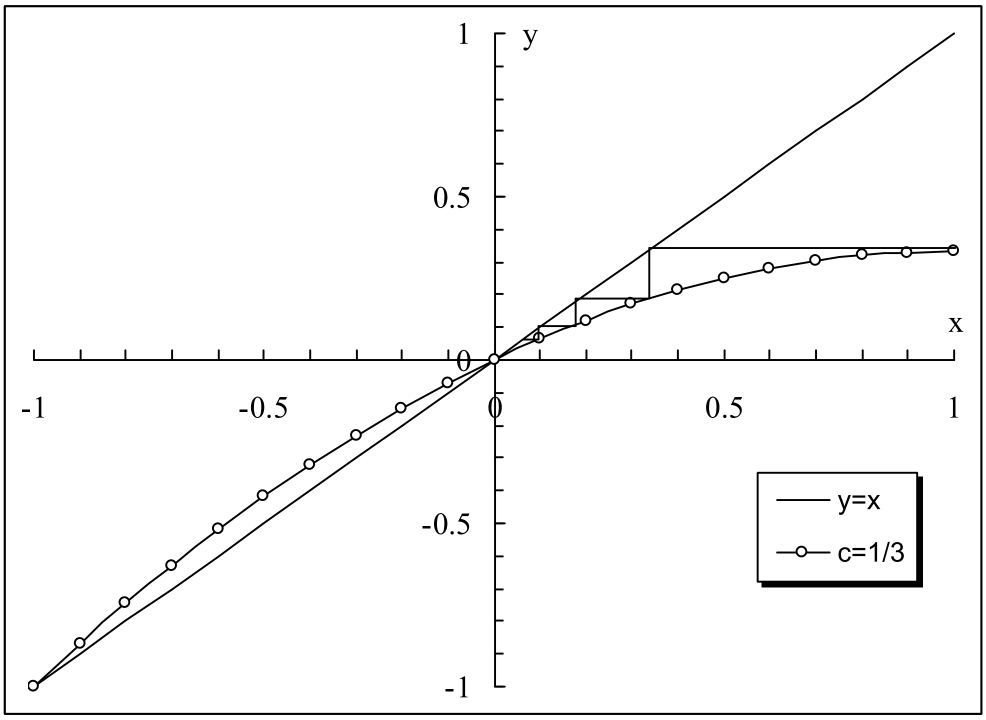
From the above analysis, we immediately draw the conclusion that for a general broad category of LDPC codes, regardless of regular or irregular LDPC codes, the erasure probability threshold over a pure BEC can be determined in theory by using (7-21) and (7-23). In particular, we have the following theorem [60] with respect to the threshold of regular LDPC codes.



(a)



(b)



(c)

Fig. 7.4 Iterative process for decoding of (2, 3) regular LDPC code over a pure BEC with various initial erasure probabilities. (a) c=2/3, (b) c=1/2 and (c) c=1/3 in relation to greater, equal and less than the associated threshold $\varepsilon^*(2,3)=1/2$, respectively.

Theorem 7.1 For a (d_v , d_c) regular LDPC codes over a pure BEC, the erasure probability threshold is determined by

$$\varepsilon^*(d_v,d_c)=\frac{1}{d_c-1} \qquad \text{(7-26a)}$$

for d_v=2. If $d_v\geq3$, the threshold is

$$\varepsilon^*(d_v,d_c)=\frac{1-\sigma}{\left(1-\sigma^{d_c-1}\right)^{d_v-1}} \tag{7-26b}$$

where σ is the unique positive real root in (0, 1) of the polynomial given by

$$q(x)=((d_v-1)(d_c-1)-1)x^{d_c-2}-\sum_{i=0}^{d_c-3}x^i \tag{7-27}$$

The proof of this theorem can be referred to the Appendix G. According to the theorem, the erasure probability threshold ε^* is immediately obtain as 1/2 for a (2, 3) regular LDPC code by using (7-26a), which agrees with the results from the example 7.3.

Example 7.4: Consider a (3, 6) regular codes ensemble, by Theorem 7.1 the erasure probability threshold is evaluated as

$$\varepsilon^*(3,6)=\frac{1-\sigma}{\left(1-\sigma^5\right)^2} \tag{7-28}$$

where σ is the unique positive real root in (0,1) of the polynomial $q(x)$ given by

$$\begin{aligned} q(x)&=((d_v-1)(d_c-1)-1)x^{d_c-2}-\sum_{i=0}^{d_c-3}x^i \\ &=(2\times5-1)\,x^4-(1+x+x^2+x^3) \\ &=9\,x^4-x^3-x^2-x-1 \end{aligned} \tag{7-29}$$

The exact value of σ is [60][63][64]

$$\sigma=\frac{1}{36}+\frac{\sqrt{(25/324)-a+b}}{2}$$

$$+\frac{\sqrt{(25/162)+a-b+\frac{685}{2916\sqrt{(25/324)-a+b}}}}{2} \tag{7-30}$$

where a and b are

$$a=\frac{22}{27}5^{2/3}\left(\frac{2}{-85+3\sqrt{24465}}\right)^{1/3} \tag{7-31a}$$

and

$$b=\frac{1}{27}\left(\frac{5}{2}\left(-85+3\sqrt{24465}\right)\right)^{1/3} \tag{7-31b}$$

Applying the results in (7-30) and (7-31) to (7-28), the erasure probability threshold of a (3, 6) regular codes ensemble can be exactly derived and approximated as

$$\varepsilon^*(3,6)=\frac{1-\sigma}{\left(1-\sigma^5\right)^2}\approx 0.42944 \tag{7-32}$$

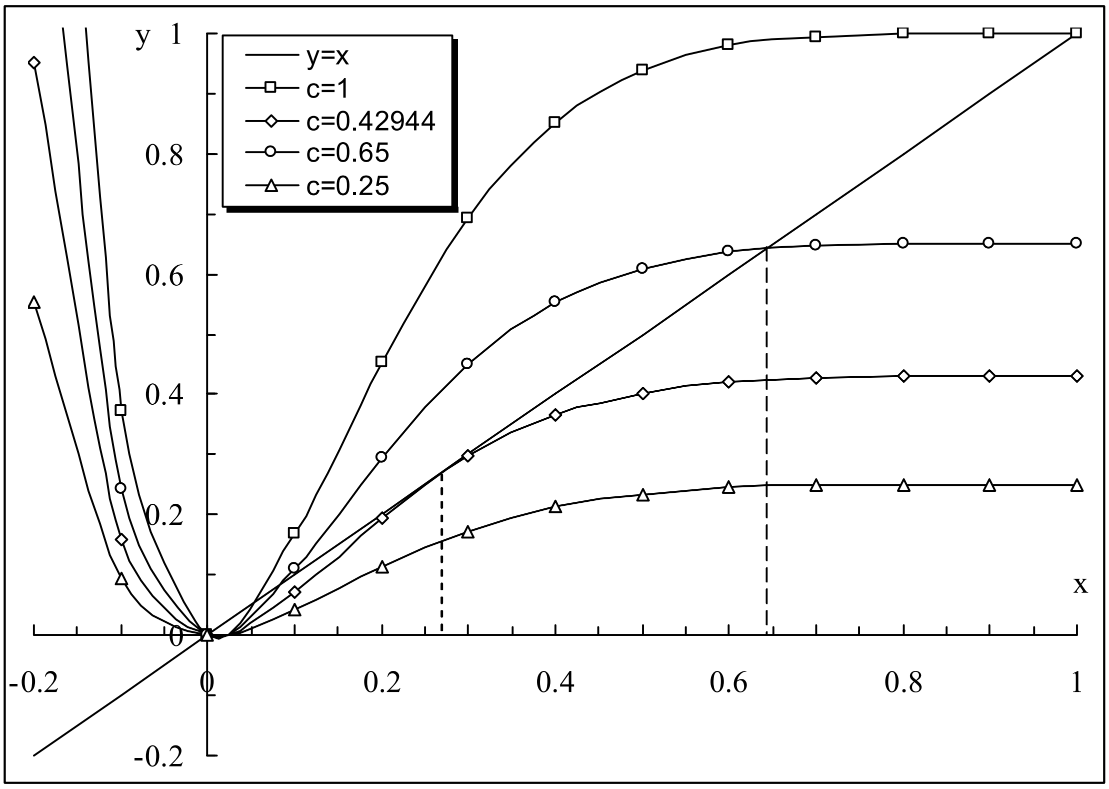


Fig. 7.5 The function $h(x)$ with c=0.65, 0.42944 and 0.25 as various initial erasure probabilities that represent the cases of greater, equal and less than the approximated threshold $\varepsilon^*(3,6)$=0.42944, respectively, and $h(x)$ with c=1 and $y=x$ are also depicted for comparisons.

Similarly, we express the function $h(x)$ for the (3, 6) regular LDPC codes ensemble as

$$h(x)=c\lambda(1-\rho(1-x))$$

$$=c\left(1-(1-x)^5\right)^2 \qquad (7\text{-}33)$$

where the coefficient c takes on the value in (0, 1). Similar as in Fig. 7.4, Fig. 7.5 describes the function $h(x)$ with c as 0.65, 0.42944 and 0.25 that stands for the cases of initial erasure probability ε greater, equal and less than the related threshold $\varepsilon^*(3,6)\approx$

0.42944, respectively. The function $h(x)$ with $c=1$ and the straight line $y=x$ are also depicted for the purpose of comparison. Fig. 7.5 shows that $y=x$ intersects the function $h(x)$ with $c=0.65$ at two points that fall into the interval (0, 1) as well as the trivial intersection point at $x=0$ for $h(x)$ with all c's. The asymptotic erasure probability $p_{erasure}^{(\infty)}$ converges to the intersection point with the largest abscissa that is obviously a nonzero value associated with the long dashed line in the figure. The line $y=x$ is tangent to $h(x)$ with $c=\varepsilon^*(3,6)$ at a point in (0, 1), which is indicated by the dashed line as shown in the figure. Obviously, $p_{erasure}^{(\infty)}$ can asymptotically converge to zero if the initial erasure probability ε is equal to $\varepsilon^*(3,6)-\varepsilon_0$, where ε_0 is a sufficiently small positive number. For $h(x)$ with $c=0.25$, the two functions has only one intersection point at $x=0$ that exactly corresponds to the convergent point $p_{erasure}^{(\infty)}$ in this case. □

7.3 Analyzing Irregular LDPC Codes from Gallager's Decoding Algorithm A over a BSC

7.3.1 Basic Definitions and Theorems for Bit Error Probability Thresholds

Recall that the basic principles described in Section 5.3 for a BSC with crossover probability p_0, if an arbitrary variable node v_n initially receives an error bit from the channel with a probability of p_0, then the probability, which an incident check node c_m is unsatisfied, is equivalent to that of an even number of the associated variable nodes other than v_n sending c_m the wrong bits.

For a (d_v, d_c) regular LDPC code, it is well known that this probability is the same for any check node incident to the variable node v_n because all the check nodes have the same degree, i.e.,

$$\frac{1+(1-2p_0)^{d_c}}{2} \tag{7-34}$$

For an irregular LDPC code with a degree distribution pair $(\lambda(x), \rho(x))$ for variable and check nodes, if a check node c_m incident to the variable node has degree i ($2 \le i \le d_c$), then the probability in (7-34) should be slightly modified as

$$\frac{1+(1-2p_0)^{i}}{2} \tag{7-35}$$

For an arbitrary check node with various possible degrees, the average probability is derived similarly as that for the BEC by taking each possible degree i with respect to the fraction (probability) ρ_i into account. Thus, we get

$$\begin{aligned}\sum_{i=2}^{d_c} \rho_i \frac{1+(1-2p_0)^{i-1}}{2} &= \frac{1}{2}\left[\sum_{i=2}^{d_c} \rho_i + \sum_{i=2}^{d_c} \rho_i (1-2p_0)^{i-1}\right] \\ &= \frac{1+\rho(1-2p_0)}{2} \end{aligned} \tag{7-36}$$

According to Gallager's decoding algorithm A, an error is successfully corrected if all the $j-1$ check nodes incident to the variable node v_n of degree j are unsatisfied. Hence, in the first decoding iteration the probability that an error bit is initially received by v_n and finally corrected by its incident check nodes is

$$p_0\left[\frac{1+\rho(1-2p_0)}{2}\right]^{j-1} \tag{7-37}$$

The average probability of this error event for an irregular LDPC code is

$$p_0\sum_{j=2}^{d_v}\lambda_j\left[\frac{1+\rho(1-2p_0)}{2}\right]^{j-1}=p_0\lambda\left(\frac{1+\rho(1-2p_0)}{2}\right) \tag{7-38}$$

If the variable node v_n initially receives a correct bit from BSC output, then an incident check node c_m is unsatisfied if and only if an odd number of variable nodes other v_n sending c_m the error bits. Hence, the average probability, which a bit is first correctly received but decoded incorrectly because of $j-1$ unsatisfied check nodes, is expressed as follows for the first decoding iteration.

$$(1-p_0)\sum_{j=2}^{d_v}\lambda_j\left[\frac{1-\rho(1-2p_0)}{2}\right]^{j-1}=(1-p_0)\lambda\left(\frac{1-\rho(1-2p_0)}{2}\right) \tag{7-39}$$

Let $q^+(y)$ and $q^-(y)$ be defined as

$$q^+(y)=\lambda\left(\frac{1+\rho(1-2y)}{2}\right) \tag{7-40a}$$

and

$$q^-(y)=\lambda\left(\frac{1-\rho(1-2y)}{2}\right) \tag{7-40b}$$

Apparently, $q^+(0)=1$ and $q^-(0)=0$ since $\rho(0)=\lambda(0)=0$ and $\rho(1)=\lambda(1)=1$. For an irregular LDPC code with a degree distribution pair $(\lambda(x),\rho(x))$ the bit error probability in this first iteration is

$$p_1 = p_0 - p_0 q^+(p_0) + (1-p_0) q^-(p_0) \tag{7-41}$$

Generically, by recursion the bit error probability in the lth (l=1, 2, …..) iteration is immediately given by

$$p_l = p_0 - p_0 q^+(p_{l-1}) + (1-p_0) q^-(p_{l-1}) \tag{7-42}$$

Let us define the function $f(x,y)$ for $0 \le x \le 1$ and $0 \le y \le 1/2$ as

$$f(x,y) = x - xq^+(y) + (1-x) q^-(y) \tag{7-43}$$

For small y, $q^+(y)$ and $q^-(y)$ can be expanded at y=0 by using the well known Maclaurin series

$$q^+(y) = \lambda(1) + \lambda'(1) \left(\frac{1+\rho(1-2y)}{2}\right)'\Bigg|_{y=0} y + O(y^2)$$

$$= 1 + \lambda'(1) \frac{(-2)\rho'(1)}{2} y + O(y^2)$$

$$= 1 - \lambda'(1)\, \rho'(1) y + O(y^2) \tag{7-44a}$$

and

$$q^-(y) = \lambda(0) + \lambda'(0) \left(\frac{1-\rho(1-2y)}{2}\right)'\Bigg|_{y=0} y + O(y^2)$$

$$= -\lambda'(0) \frac{(-2)\rho'(1)}{2} y + O(y^2)$$

$$= \lambda_2 \rho'(1) y + O(y^2) \tag{7-44b}$$

where $O(y^2)$ notation means that there exists positive constants r_c and δ independent of y such that $\left|O(y^2)/y^2\right|<r_c$ for all $|y|<\delta$, $\lambda'(y)$ and $\rho'(y)$ denote derivatives of $\lambda(y)$ and $\rho(y)$ with respect to y, respectively. Applying (7-44a) and (7-44b) to (7-43), we have $f(x,y)$ for small y conditioned on a fixed x

$$\begin{aligned} f(x,y) &= x-x\left(1-\lambda'(1)\rho'(1)y+O(y^2)\right)+(1-x)\left(\lambda_2\rho'(1)y+O(y^2)\right) \\ &= x\left(1-1+\lambda'(1)\rho'(1)y-\lambda_2\rho'(1)y\right)+\lambda_2\rho'(1)y-xO(y^2)+(1-x)O(y^2) \\ &= x\left(\lambda'(1)\rho'(1)-\lambda_2\rho'(1)\right)y+\lambda_2\rho'(1)y+O(y^2) \qquad (7\text{-}45) \end{aligned}$$

The function $f(x,y)$ has some important properties described by the following theorem. Note that in the analysis of irregular LDPC codes over a BSC the stability condition [9] $\lambda_2\,\rho'(1)<1$ is always fulfilled. For more detailed analysis on the general stability issues for LDPC codes over various noise channels, the interested readers can refer to [8][9].

Theorem 7.2: The function $f(x,y)$ is an increasing function in both its arguments in the range $0\le x\le 1$ and $0<y\le 1/2$. Furthmore, if

$$x<\frac{1-\lambda_2\rho'(1)}{\lambda'(1)\rho'(1)-\lambda_2\rho'(1)} \qquad (7\text{-}46)$$

then, for small enough y, we have $f(x,y)<y$. □

The proof of the theorem can be referred to the Appendix H. Similar as the error probability threshold ε^* for regular LDPC codes over BSC, in this section we will generalize the basic notions to irregular LDPC codes, which clearly defines the threshold ε^* as

Definition 7.3: The bit error probability threshold ε^* for an irregular LDPC codes ensemble using Gallager's decoding algorithm A over a BSC is the supremum of all p_0 in [0, 1/2] such that p_l as defined in (7-42) converges to zero as l tends to infinity. □

Similar as the analysis in BEC, it follows that for any $0 \le p_0 < \varepsilon^*$ the sequence p_l will converge to zero. Moreover, p_l must be strictly decreasing with the increasing l. It would be more convenient to work with an alternative description of the threshold described by the following definition.

Definition 7.4: The threshold ε_0^* is the supremum of all p_0 in [0, 1/2] such that

$$x = p_0 - p_0 q^+(x) + (1 - p_0) q^-(x) \tag{7-47}$$

does not have a strictly *positive* solution x with $x \le p_0$. □

The next theorem further shows that the thresholds ε^* and ε_0^* are exactly equal under the stability condition $\lambda_2 \rho'(1) < 1$. From definitions 7.3 and 7.4 we have

Theorem 7.3: If $\lambda_2 \rho'(1) < 1$ then $\varepsilon^* = \varepsilon_0^*$.

The proof for the theorem can be referred to the Appendix I. According to the theorem the bit error rate threshold $\varepsilon^*(\lambda, \rho)$ can be alternatively determined by the definition 7.4 rather than finding it directly by the original definition 7.3. This implies that $\varepsilon^*(\lambda, \rho)$ is the minimum value of all p_0 in [0, 1/2] such that the equation in (7-47) has at least one strictly position solution x with $x \le p_0$. Therefore, p_0 is a function of x, which is denoted by $g(x)$ and expressed as

$$g(x) = \frac{x - q^-(x)}{1 - q^+(x) - q^-(x)} \tag{7-48}$$

Applying l'Hôpital's rule, the limit of $g(x)$ also exists as x approaches to zero and $g(0)$ can be defined by

$$\begin{aligned} g(0) &= \lim_{x \to 0} \frac{\left(x - q^-(x)\right)'}{\left(1 - q^+(x) - q^-(x)\right)'} \\ &= \frac{1 - \lambda_2 \rho'(1)}{(\lambda'(1) - \lambda_2)\rho'(1)} > 0 \end{aligned} \tag{7-49}$$

Hence, $g(x)$ is well defined and continuous on [0, 1/2]. Theoretically, similar as that for BEC the threshold $\varepsilon^*(\lambda, \rho)$ for BSC is specified as

$$\varepsilon^*(\lambda, \rho) = \min_{0 \le x \le 1/2} \{g(x) : g(x) \ge x\} \tag{7-50}$$

Furthermore,

$$\begin{aligned} g(1/2) &= \frac{1/2 - q^-(1/2)}{1 - q^+(1/2) - q^-(1/2)} \\ &= \frac{1/2 - \lambda(1/2)}{1 - 2\lambda(1/2)} = \frac{1}{2} \end{aligned} \tag{7-51}$$

The above equation implies that the straight line $y = x$ intersects $g(x)$, at least, one intersection point at x=1/2 corresponding to $g(x)$=1/2. Let us investigate the equation $g(x) = x$, i.e.,

$$\frac{x-q^{-}(x)}{1-q^{+}(x)-q^{-}(x)}=x \tag{7-52a}$$

Equivalently,

$$xq^{+}(x)+(x-1)q^{-}(x)=0 \tag{7-52b}$$

Let $s(x)$ denote the polynomial $xq^{+}(x)+(x-1)q^{-}(x)$. Hence, finding all the intersection points for $y=x$ and $g(x)$ is equivalently to finding all the solutions x's such that $s(x)=0$. Let τ be the small positive root of $s(x)$ in (0, 1/2), this immediately gives rise to an upper bound on the bit error probability threshold for an irregular LDPC code over a BSC as

$$\begin{aligned}\varepsilon^{*}(\lambda,\rho)&\le\min\{g(0),\tau\}\\ &=\min\left\{\frac{1-\lambda_2\rho'(1)}{\lambda'(1)\rho'(1)-\lambda_2\rho'(1)},\tau\right\}\end{aligned} \tag{7-53}$$

To determine the bit error probability threshold $\varepsilon^{*}(\lambda,\rho)$, first we should specify the upper bound of the threshold by (7-53), then decide if the upper bound is exactly the threshold. Otherwise, we have to find out the minimum value of $g(x)$ over the interval [0, 1/2]. The following example will give more details on the determination.

7.3.2 An Example of Bit Error Probability Threshold for (3, 6) Regular LDPC Codes Ensemble

Example 7.5: We will still consider the (3, 6) regular LDPC codes ensemble in this example, whose degree distributions for variable and check nodes are $\lambda(x)=x^2$ and $\rho(x)=x^5$, respectively. Here $q^{+}(x)$ and $q^{-}(x)$ are

$$q^{+}(x)=\lambda\left(\frac{1+\rho(1-2x)}{2}\right)=\frac{1}{4}\left(1+(1-2x)^{5}\right)^{2} \tag{7-54a}$$

and

$$q^{-}(x)=\lambda\left(\frac{1-\rho(1-2x)}{2}\right)=\frac{1}{4}\left(1-(1-2x)^{5}\right)^{2} \tag{7-54b}$$

Hence, $s(x)$ is

$$\begin{aligned} s(x)&=xq^{+}(x)+(x-1)q^{-}(x)\\ &=\frac{1}{4}x\left(1+(1-2x)^{5}\right)^{2}+\frac{1}{4}(x-1)\left(1-(1-2x)^{5}\right)^{2} \end{aligned} \tag{7-55}$$

Using $(1-x)/2$ to replace x for $s(x)$, we get

$$\begin{aligned} s\left(\frac{1-x}{2}\right)&=\frac{1}{8}(1-x)\left(1+x^{5}\right)^{2}-\frac{1}{8}(1+x)\left(1-x^{5}\right)^{2}\\ &=\frac{1}{4}(1-x)x(1+x)\left(-1-x^{2}+x^{4}+x^{6}+x^{8}\right)\\ &=\frac{1}{4}(1-x)x(1+x)\,r\left(x^{2}\right) \end{aligned} \tag{7-56}$$

where $r(x)$=$-1-x+x^{2}+x^{3}+x^{4}$. Apparently, x=1, 0 and –1 are the roots of $s\left(\frac{1-x}{2}\right)$=0, corresponding to the roots x=0, 1/2 and 1 of $s(x)$=0, which are beyond the interval (0, 1/2). Thus, we only focus on the polynomial $r(x)$ that fulfills $r(0)$=–1<0 and $r(1)$=1>0, which implies that $s\left(\frac{1-x}{2}\right)$=0 has at least one positive root in (0, 1). On the other hand, applying Descartes' rule of signs [63][64], it is easily to see that $s\left(\frac{1-x}{2}\right)$ has at most one positive root over the entire real number field. Thus, $s\left(\frac{1-x}{2}\right)$ must has a single root denoted by σ in (0, 1), which is associated with the unique root of the polynomial $s(x)$ as

$$\tau=(1-\sqrt{\sigma})/2 \tag{7-57}$$

The exact value of σ is obtained by [60][63][64]

$$\sigma=-\frac{1}{4}+\frac{\sqrt{(-5/12)-\eta}}{2}$$
$$+\frac{\sqrt{(-5/6)+11/\left(4\sqrt{(-5/12)-\eta}\right)+\eta}}{2} \tag{7-58a}$$

where

$$\eta=\frac{8}{3}\left(\frac{2}{83+3\sqrt{993}}\right)^{1/3}-\frac{1}{3}\left(\frac{83+3\sqrt{993}}{2}\right)^{1/3} \tag{7-58b}$$

Numerically, the root τ in terms of first five digits [60] is approximated as $\tau\approx0.03946$. On the other hand, $g(0)$ as evaluated by (7-49) is 0.1 that is much greater than τ. Thus, the threshold $\varepsilon^*(\lambda,\rho)$ is upper bounded by

$$\varepsilon^*(3,6)\le\tau\approx0.03946 \tag{7-59}$$

Then we will determine the threshold $\varepsilon^*(3,6)$ by finding the minimum value of $g(x)$ that fulfills the requirement in (7-50). Applying (7-48) and (7-54a) and (7-54b), the function $g(x)$ for the (3, 6) regular LDPC code in this example is

$$g(x) = \frac{x - q^{-}(x)}{1 - q^{+}(x) - q^{-}(x)}$$

$$= \frac{4x - \left(1 - (1-2x)^5\right)^2}{4 - \left(1 + (1-2x)^5\right)^2 - \left(1 - (1-2x)^5\right)^2} \tag{7-60}$$

whose curve is depicted in Fig. 7.6. We can easily observe that the straight line $y=x$ intersects the $g(x)$ curve at about x=0.40 that is the only intersection point except the trivial one at x=1 and indicated by the dashed line. Obviously, it has the smallest abscissa that fulfills $g(x) \geq x$ over the interval [0, 0.5]. Thus, the obtained value 0.03946 is verified to be the approximated bit error rate threshold $\varepsilon^{*}(3,6)$.

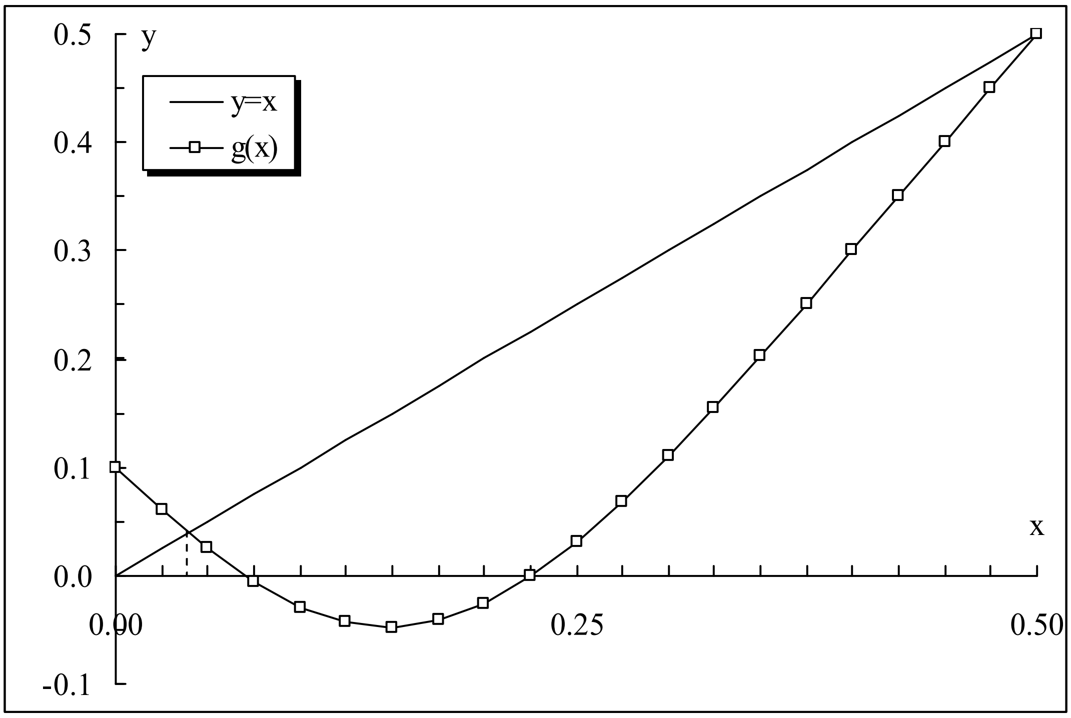


Fig. 7.6 The $g(x)$ curve for a (3, 6) regular LDPC codes ensemble over a BSC with the point intersected by the straight line $y=x$ at about x=0.4.

Similar as that for BEC, we define the function $h(x)$ for irregular LDPC codes ensemble in BSC as

$$h(x)=c-cq^{+}(x)+(1-c)q^{-}(x) \tag{7-61}$$

where $c\in[0,\ 0.5]$ represents the initial bit error probability (or crossover probability of a BSC). Fig. 7.7 shows the $h(x)$ curves, intersected by the straight line $y=x$, with $c=$ 0.030, 0.03946 and 0.045 that are greater, equal and less than the associated threshold $\varepsilon^{*}(3,6)\approx 0.03946$, respectively.

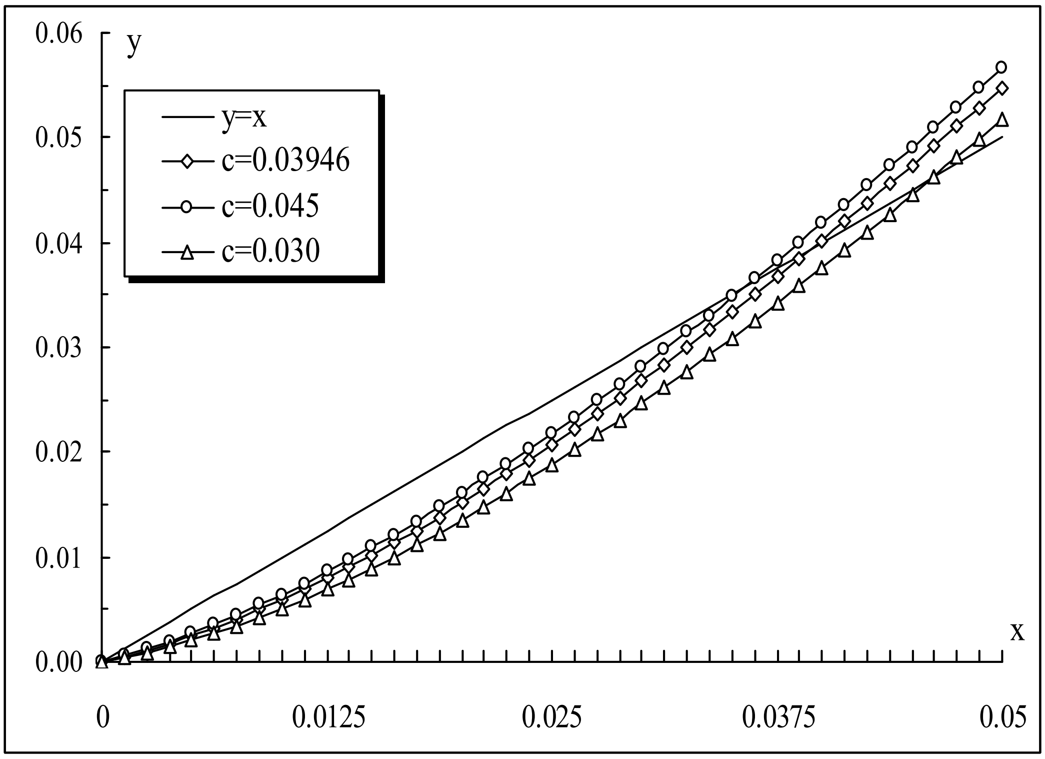
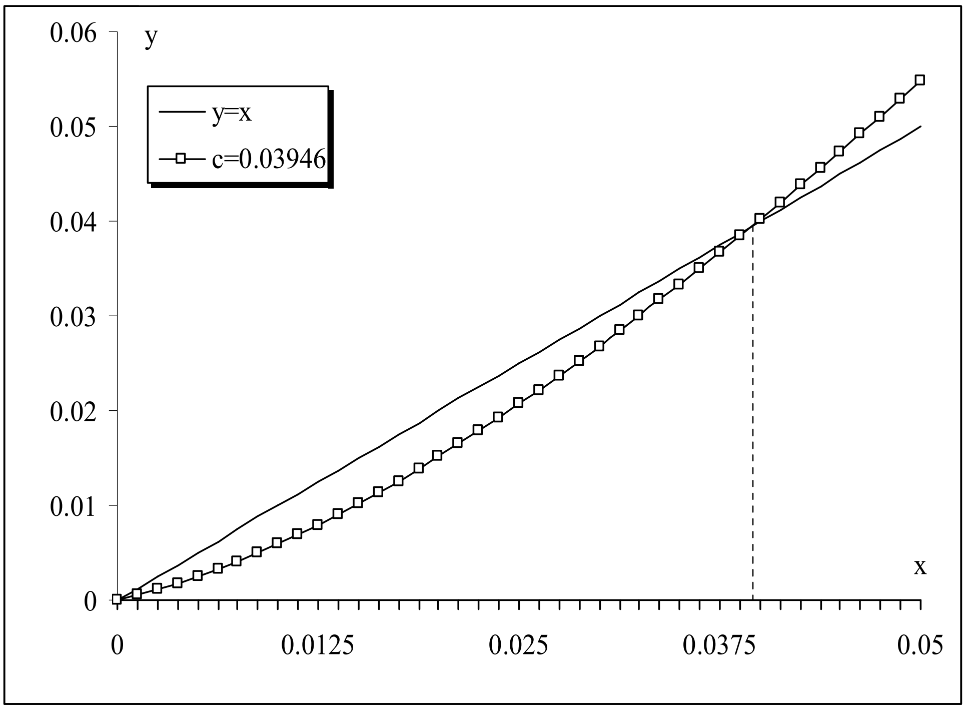
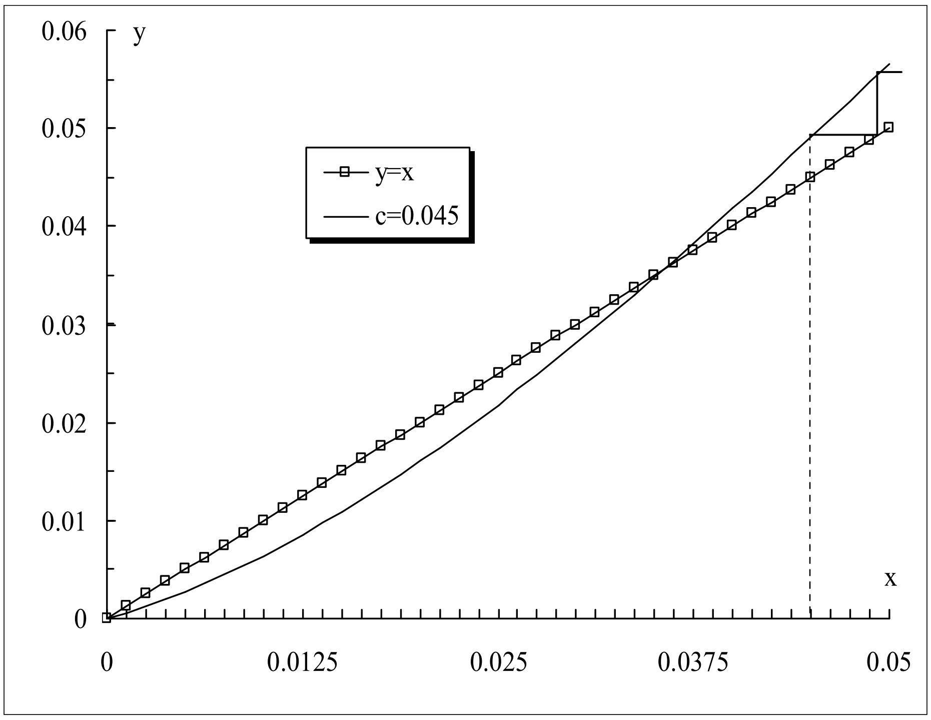


Fig. 7.7 The function $h(x)$ with c=0.045, 0.03946 and 0.030 as various initial bit error probabilities that are greater, equal and less than the approximated threshold $\varepsilon^{*}(3,6)\approx$0.03946, respectively.

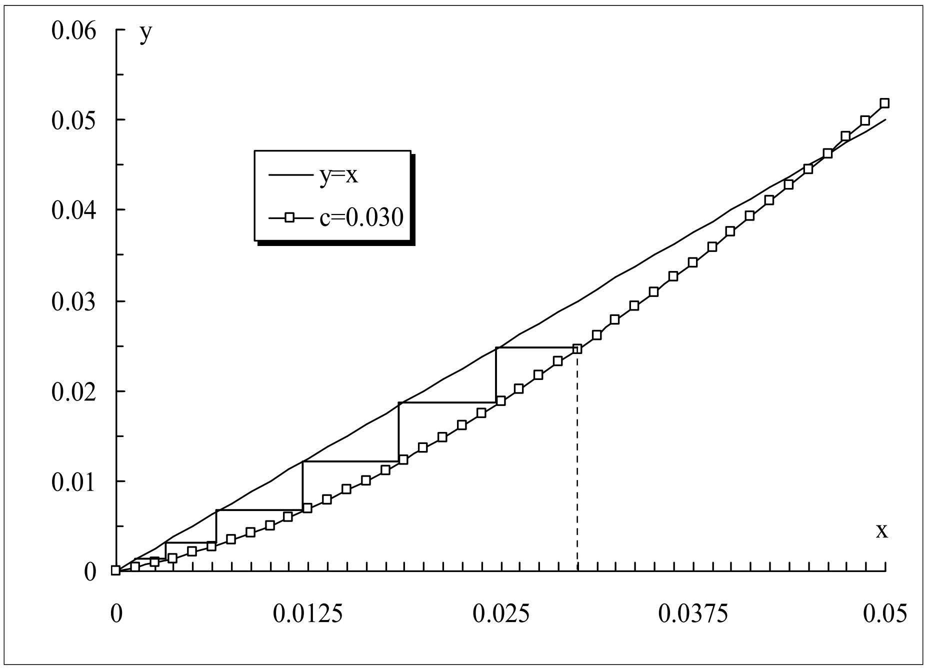
It can be clearly observed that for $h(x)$ with c=0.03946 the abscissa of the nonzero point intersected by $y=x$ is about at x=0.03946, which is indicated by dashed line in Fig. 7.8(a). For c greater than 0.03946, i.e., 0.045 in this example, the abscissa of the intersection point is approximately at x=0.0363 that is less than 0.045. We can easily see that the bit error probability gets larger with the decoding iterations and cannot converge to zero with initial error probability p_0=0.045, which is illustrated in Fig. 7.8(b) with p_0 in relation to the dashed line. For c less that 0.03946, i.e., 0.030 in this example, the abscissa of the intersection point is about at x=0.0463 that is greater than 0.030. We can also observe that the bit error probability gets smaller with the decoding iterations and gradually converges to zero with initial error probability p_0=0.030, whose convergent process is illustrated in Fig. 7.8(c) with p_0 associated with the dashed line.



(a) $p_0=\varepsilon^*(3,6)$



(b) $p_0 > \varepsilon^*(3,6)$



(c) $p_0 < \varepsilon^*(3,6)$

Fig. 7.8 The iterative process for decoding of a (3, 6) regular LDPC code with initial bit error probability (a) p_0=0.03946$\approx \varepsilon^*(3,6)$ (b) p_0=0.045$> \varepsilon^*(3,6)$ and (c) p_0=0.030$< \varepsilon^*(3,6)$, which are indicated by their respective dashed lines.

Chapter 8 Irregular Low-Density Parity-Check Codes: Optimal Design by Linear and Nonlinear Programming

8.1 Design of Optimized Irregular LDPC Codes Based on Gallager's Decoding Algorithm A over BSC

After analyzing the thresholds of generic irregular LDPC codes decoded by Gallager's decoding algorithms an over BEC and BSC channels, it is quite natural that one might wonder whether optimal degree distributions for variable and check nodes for the given decoder can be effectively found. We will show in this section that such optimal distributions can be obtained in the form of concentrated distributions, i.e., degree distribution pairs for which variable node degree distribution (or check node degree distribution) is reduced to at most two nonzero degrees and for which these nonzero degrees are consecutive.

The distribution $\lambda(x)$ for variable nodes in this form is called *left-concentrated* degree distribution denoted by $\lambda^c(x)$ if it can be expressed as

$$\lambda^c(x)=\lambda_i^c x^{i-1}+\lambda_{i+1}^c x^i \quad (i=2,3\ldots.) \tag{8-1a}$$

Similarly, the distribution $\rho(x)$ for check nodes in this form is called *right-concentrated* degree distribution denoted by $\rho^c(x)$ if it can be expressed as

$$\rho^c(x)=\rho_j^c x^{j-1}+\rho_{j+1}^c x^j \quad (j=2,3\ldots.) \tag{8-1b}$$

The concentrated distributions $\lambda^c(x)$ and $\rho^c(x)$ can be proved to be optimal by using the basic principle of linear programming [64][65]. Given a *primal* and a *dual* problem and a *feasible* solution for each problem, which is optimal if the values of both objective (or cost) functions are exactly the same. The fundamentals of the linear programming and related conclusions can be referred to the Appendix *K*.

We start our optimization procedure by first introducing the so-called average degrees for variable and check nodes given the general degree distribution pair ($\lambda(x), \rho(x)$). For variable node with the number of edges *E* emanating from all variable nodes in a bipartite graph of an irregular LDPC code, the number of a variable node of degree i is $E\lambda_i / i$ and the total number of the variable nodes is $\sum_{i=2}^{+\infty} E\lambda_i / i$. Thus, the average degree of an arbitrary variable node of an irregular LDPC code is obtained

$$\frac{E}{\sum_{i=2}^{+\infty}(E\lambda_i)/i} = \frac{1}{\sum_{i=2}^{+\infty}\frac{\lambda_i}{i}} = \frac{1}{\int_0^1 \lambda(x)dx} \tag{8-2a}$$

Let $\int\lambda$ denote the integer $\int_0^1 \lambda(x)dx$, hence $\left(\int\lambda\right)^{-1}$ is simply expressed as the average degree of a variable node.

Example 8.1: An irregular LDPC code has a graphical representation in which half the edges emanating from the variable nodes of degree 2 and half the edges from the variables nodes of degree 3, i.e., $\lambda(x)=\frac{1}{2}x+\frac{1}{2}x^2$. Thus, $\int\lambda=5/12$ and the average degree of a variable node is $\left(\int\lambda\right)^{-1}=12/5$. □

Similarly, let $\int\rho$ denote the integer $\int_0^1\rho(x)dx$. Thus the average degree of a check node is

$$\frac{E}{\sum_{i=2}^{+\infty}(E\rho_i)/i}=\frac{1}{\sum_{i=2}^{+\infty}\frac{\rho_i}{i}}$$

$$=\frac{1}{\int_0^1\rho(x)dx}=\left(\int\rho\right)^{-1} \tag{8-2b}$$

The notion of average degree of a check (or variable) node is very important for the subsequent analysis in determining the optimal left-concentrated degree distribution.

8.2 The Optimal Left-Concentrated Degree Distribution

In this section, we will focus on the optimization concerning the degree distribution of a variable node for an irregular LDPC code over a BSC. The optimized solution will be proved in the form of the left-concentrated degree distribution that can results in a large bit error rate threshold in contrast to the distribution without optimization.

Let us revisit the results in the preceding analysis that the threshold $\varepsilon^*(\lambda,\rho)$ for an arbitrary irregular LDPC code with degree distribution pair ($\lambda(x),\rho(x)$) over a BSC is upper bounded by

$$\varepsilon^*(\lambda,\rho)\le\min\{g(0),\tau\} \tag{8-3}$$

where $g(0)$ is given by (7-49) based on $g(x)$ defined by (7-48) as

$$g(0)=\frac{1-\lambda_2\rho'(1)}{\lambda'(1)\rho'(1)-\lambda_2\rho'(1)} \tag{8-4}$$

and the parameter τ is the smallest positive root of the polynomial $s(x)=xq^+(x)+(x-1)q^-(x)$ over the interval (0, 1/2).

We will first show in the sequel that any concentrated degree distribution on $\lambda(x)$ can increase the value of $g(0)$. Second, the optimal left-concentrated degree distribution $\lambda^c(x)$ can be found on the basis of maximizing τ by applying the nonlinear and the linear program, which subsequently increases the value of $g(0)$. By (8-3) this optimized left-concentrated degree distribution may result in a larger threshold $\varepsilon^*(\lambda^c,\rho)$ than $\varepsilon^*(\lambda,\rho)$ without concentration on $\lambda(x)$. Finally, as a consequence, the optimization on the irregular LDPC code design will definitely result in lower bit error rate over a BSC using Gallager's decoding algorithm A.

8.2.1 The impact of concentrated degree distribution on the quantity of $g(0)$

In this part we will carefully investigate whether the quantity $g(0)$ is increased or not by concentration on the degree distribution $\lambda(x)$ for variable nodes. The next theorem clearly states the fact that the value $g(0)$ is really enlarged by the aforementioned concentration with given $\int\lambda$ and $\rho'(1)$.

Theorem 8.1 For the fixed $\int\lambda$ and $\rho'(1)$ for variable and check nodes, respective, the quantity $g(0)=\dfrac{1-\lambda_2\rho'(1)}{\lambda'(1)\rho'(1)-\lambda_2\rho'(1)}$ can be increased by a concentrated degree distribution of variable nodes.

The proof for the theorem can be referred to the Appendix I, which clearly indicates that a concentrated degree distribution can increases the quantity of $g(0)$.

Example 8.2: Consider an irregular LDPC codes ensemble with degree distribution pair $(\lambda(x),\rho(x))$, where $\lambda(x)=\frac{1}{4}x+\frac{1}{2}x^2+\frac{1}{4}x^3$ and $\rho(x)=x^3$. Thus, $\lambda'(1)$=2, $\lambda_2=\frac{1}{4}$ and $\rho'(1)$=3, and satisfying the stability condition $\lambda_2\rho'(1)=\frac{3}{4}$<1. The value of $g(0)$ is

$$g(0)=\frac{1-\lambda_2\rho'(1)}{\lambda'(1)\rho'(1)-\lambda_2\rho'(1)}$$

$$=\frac{1-\frac{1}{4}\times 3}{2\times 3-\frac{1}{4}\times 3}=\frac{1}{21} \tag{8-5}$$

Based on the same method used to prove the Theorem 8.1 in the Appendix I, let i=2, j=3 and k=4, and we define the following variables

$$\eta_i=-(k-j)i=-2 \tag{8-6a}$$

$$\eta_j=(k-i)j=6 \tag{8-6b}$$

$$\eta_k=-(j-i)k=-4 \tag{8-6c}$$

Consider the polynomial with δ=1/32, which is given as

$$\lambda^{\delta}(x) = \lambda(x) + \delta(\eta_i x^{i-1} + \eta_j x^{j-1} + \eta_k x^{k-1})$$

$$= \frac{3}{16}x + \frac{11}{16}x^2 + \frac{1}{8}x^3 \qquad (8\text{-}7)$$

Clearly, $\sum_{l=2}^{4}\lambda_l^{\delta}$ =1, $\int_0^1 \lambda^{\delta}(x)dx = \int_0^1 \lambda(x)dx$ =17/48 and $\frac{d}{dx}\lambda^{\delta}(x)\Big|_{x=1}$ =31/16. The well-known stability condition $\lambda_2^{\delta}\rho'(1) = \frac{9}{16}$<1 is also fulfilled for the concentrated degree distribution pair ($\lambda^{\delta}(x), \rho(x)$). It is easy to observe that $\lambda^{\delta}(x)$ is concentrated concerning with λ_3^{δ}, whose value is increased from the original λ_3=1/2 in $\lambda(x)$ to λ_3^{δ}=11/16 in $\lambda^{\delta}(x)$ after the concentration processing, while its neighbors λ_2^{δ} and λ_4^{δ} are correspondingly reduced to 3/16 and 1/8 in comparison with the original values $\lambda_2 = \lambda_4$=1/4 in $\lambda(x)$, respectively. Hence, the value of $g(0)$ regarding the concentrated distribution pair ($\lambda^{\delta}(x)$, $\rho(x)$) is

$$g(0) = \frac{1-\lambda_2^{\delta}\rho'(1)}{(\lambda^{\delta}(x))'\big|_{x=1}\rho'(1) - \lambda_2^{\delta}\rho'(1)}$$

$$= \frac{1-\frac{3}{16}\times 3}{\frac{31}{16}\times 3 - \frac{3}{16}\times 3} = \frac{1}{12} \qquad (8\text{-}8)$$

which is greater than 1/21 evaluated by (8-5) corresponding to the degree distribution pair ($\lambda(x), \rho(x)$). □

8.2.2 Finding the optimal left-concentrated degree distribution $\lambda^{c}(x)$ related to τ

8.2.2.1 Establish the nonlinear and linear programming problems

We have known in the preceding analysis that a concentrated degree distribution for variable node can increase the quantity of $g(0)$. In the subsequent part we will find an optimized left-concentrated degree distribution $\lambda^c(x)$ that can increase the value of τ, which is the smallest positive root of the polynomial $s(x)=xq^+(x)+(x-1)q^-(x)$ over interior (0, 1/2) and in relation to the threshold $\varepsilon^*(\lambda,\rho)$ by (8-3). It is easy to check that $s(0)=0$ and

$$s'(x)=q^+(x)-x\lambda'\left(\frac{1+\rho(1-2x)}{2}\right)\rho'(1-2x)$$

$$+q^-(x)+(x-1)\lambda'\left(\frac{1-\rho(1-2x)}{2}\right)\rho'(1-2x) \tag{8-9}$$

Obviously, $s'(0)=1-\lambda_2\rho'(1)>0$ holds due to the stability condition. Hence, $s(x)$ is an increasing function and greater than zero for x small enough. That is

$$xq^+(x)+(x-1)q^-(x)>0 \tag{8-10a}$$

Equivalently, it results in

$$\frac{x}{1-x}>\frac{q^-(x)}{q^+(x)}=\frac{\sum_{i\geq 2}\lambda_i u^{i-1}}{\sum_{i\geq 2}\lambda_i (1-u)^{i-1}} \tag{8-10b}$$

where $u=(1-\rho(1-2x))/2$. We will show in the sequel that

$$\frac{q^-(x)}{q^+(x)}=\frac{\sum_{i\geq 2}\lambda_i u^{i-1}}{\sum_{i\geq 2}\lambda_i (1-u)^{i-1}} \tag{8-11}$$

is simultaneously minimized over the whole range $u \in [0,\ 1/2]$ by the unique left–concentrated degree distribution $\lambda_c(x)$ with fixed parameter $\int_0^1 \lambda(x)dx$ denoted by $\int \lambda$. For any fixed $u \in [0,\ 1/2]$, the minimization of (8-11) by optimization will result in the decreasing and increasing regarding the functions $q^-(x) = \sum_{i\geq 2}\lambda_i u^{i-1}$ and $q^+(x) = \sum_{i\geq 2}\lambda_i(1-u)^{i-1}$, respectively. This will definitely maximize the value of τ since τ is usually very small such that fulfills the condition that (8-10a) holds. Note that for u=0 and u=1/2, the values in (8-11) are 0 and 1, respectively, independent of the choice of $\lambda(x)$. Thus, we need to find an optimal solution of the nonlinear program [64][65] as

$$\min\left\{ \frac{\sum_{j\geq 2}\lambda_j u^{j-1}}{\sum_{j\geq 2}\lambda_j (1-u)^{j-1}} \middle| \lambda_j \geq 0;\ \sum_{j\geq 2}\lambda_j = 1;\ \sum_{j\geq 2}\frac{\lambda_j}{j} = \int\lambda \right\} \tag{8-12}$$

The above nonlinear program is in relation to a linear program [64][65] as follows

$$\min\left\{ \sum_{j\geq 2}\gamma_j u^{j-1} \middle| \gamma_j \geq 0;\ \sum_{j\geq 2}\gamma_j\left(\int\lambda - \frac{1}{j}\right) = 0;\ \sum_{j\geq 2}\gamma_j(1-u)^{j-1} = 1 \right\} \tag{8-13}$$

If a feasible solution [64][65] to the constraints of the linear program (8-13) is found to be $\gamma(x) = \sum_{j\geq 2}\gamma_j x^{j-1}$, then let $\lambda(x) = \gamma(x)/\left(\sum_{l\geq 2}\gamma_l\right)$ and we can easily check that $\lambda_j = \gamma_j / \sum_{l\geq 2}\gamma_l \geq 0$ and $\sum_{j\geq 2}\lambda_j = \sum_{j\geq 2}\gamma_j / \sum_{l\geq 2}\gamma_l = 1$ that agree with the first and second constraints of the nonlinear program (8-12). For the third constraint, we have the following relationship based on the second constraint of (8-13).

$$\sum_{j\geq 2}\frac{\lambda_j}{j}=\frac{\sum_{j\geq 2}\frac{\gamma_j}{j}}{\sum_{l\geq 2}\gamma_l}$$

$$=\frac{\left(\int\lambda\right)\sum_{j\geq 2}\gamma_j}{\sum_{l\geq 2}\gamma_l}=\int\lambda \qquad (8\text{-}14)$$

Thus, the polynomial $\lambda(x)$, associated with a feasible solution to the constraints of the linear program, can fulfill all the constraints in (8-12), i.e., a feasible solution to the constraints of the nonlinear program. Finally, we evaluate the value of the objective (cost) function of the linear program with $\lambda(x)$, i.e.,

$$\frac{\sum_{j\geq 2}\lambda_j u^{j-1}}{\sum_{j\geq 2}\lambda_j(1-u)^{j-1}}=\frac{\sum_{j\geq 2}\gamma_j u^{j-1}/(\sum_{l\geq 2}\gamma_l)}{\sum_{j\geq 2}\gamma_j(1-u)^{j-1}/(\sum_{l\geq 2}\gamma_l)}$$

$$=\frac{\sum_{j\geq 2}\gamma_j u^{j-1}/(\sum_{l\geq 2}\gamma_l)}{1/(\sum_{l\geq 2}\gamma_l)}=\sum_{j\geq 2}\gamma_j u^{j-1} \qquad (8\text{-}15)$$

Clearly, the value of the objective function of the nonlinear program is corresponding to a same value of the objective function of the linear program, which is related to a feasible solution. If the feasible solution to the constraints that achieves the minimum value of the object function of the linear program, then the corresponding value of the objective function of the nonlinear program is also minimized. Thus, the task of finding an optimal solution to the nonlinear program is equivalent to finding the associated optimal solution to the linear program.

8.2.2.2 Find the optimal solution to the linear program

Let us briefly revisit the linear program whose standard form is given by

$$\text{minimize} \quad \mathbf{c}^T\mathbf{x} \tag{8-16a}$$

$$\text{subject to} \quad \mathbf{Ax}=\mathbf{b} \tag{8-16b}$$

$$\mathbf{x}\geq\mathbf{0} \tag{8-16c}$$

where $\mathbf{x}$ is an n-dimensional column vector of nonnegative in each component, $\mathbf{c}^T$ is an n-dimensional row vector, $\mathbf{b}$ is an m-dimensional column vector ($m<n$), and $\mathbf{A}$ is an $m \times n$ matrix. The expression (8-16a) to be minimized is called the objective function of the linear program, and the equality and inequality in (8-16b) and (8-16c) are called the constraints of the linear program.

The dual linear program of the linear program defined by (8-16) is

$$\text{maximize} \quad \boldsymbol{\beta}^T\mathbf{b} \tag{8-17a}$$

$$\text{subject to} \quad \boldsymbol{\beta}^T\mathbf{A}\leq\mathbf{c}^T \tag{8-17b}$$

where $\boldsymbol{\beta}^T$ is an m-dimensional row vector. The vector $\mathbf{x}$ is the variable of the primal problem, and $\boldsymbol{\beta}$ is the variable of the dual problem, which is not restricted to be nonnegative. For more details the interested reader can refer to the Appendix J.

Consider the primal program (8-13) where the matrix $\mathbf{A}$ is

$$\mathbf{A}=\begin{bmatrix} \int\lambda-\frac{1}{2} & \int\lambda-\frac{1}{3} & \cdots & \int\lambda-\frac{1}{i} & \int\lambda-\frac{1}{i+1} & \cdots \\ (1-u) & (1-u)^2 & \cdots & (1-u)^{i-1} & (1-u)^i & \cdots \end{bmatrix} \tag{8-18a}$$

Suppose that the rank of $\mathbf{A}$ is 2. The vectors $\mathbf{x}$, $\mathbf{c}$, $\mathbf{b}$ are

$$\mathbf{x}^T=[\gamma_2,\ \gamma_3,\ \ldots\ldots,\ \gamma_i,\ \gamma_{i+1},\ \ldots\ldots] \tag{8-18b}$$

$$\mathbf{c}^T=[u,\ u^2,\ \ldots\ldots,\ u^{i-1},\ u^i,\ \ldots\ldots] \tag{8-18c}$$

$$\mathbf{b}^T=[0,\ 1] \tag{8-18d}$$

(i) Find a basic feasible solution to the constraints of the primal program

Let $\mathbf{B}$ be the submatrix made up of two linearly independent columns of $\mathbf{A}$, i.e.,

$$\mathbf{B}=\begin{bmatrix} \int\lambda-\frac{1}{i} & \int\lambda-\frac{1}{i+1} \\ (1-u)^{i-1} & (1-u)^i \end{bmatrix} \tag{8-19}$$

To find a basic feasible solution to the constraints of the primal program, let $i=\lfloor 1/\int\lambda \rfloor$ that is related to the average degree of a variable node. Based on Theorem 8 in the Appendix J, the two-dimensional vector $\mathbf{x}_B^T=[\gamma_i,\ \gamma_{i+1}]$ corresponding to the basis $\mathbf{B}$ is

$$\begin{aligned}\mathbf{x}_B&=\mathbf{B}^{-1}\mathbf{b}\\ &=\begin{bmatrix} \int\lambda-\frac{1}{i} & \int\lambda-\frac{1}{i+1} \\ (1-u)^{i-1} & (1-u)^i \end{bmatrix}^{-1}\begin{bmatrix}0\\1\end{bmatrix}\\ &=\frac{1}{\det(\mathbf{B})}\begin{bmatrix} (1-u)^i & -\left(\int\lambda-\frac{1}{i+1}\right) \\ -(1-u)^{i-1} & \int\lambda-\frac{1}{i} \end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix}\end{aligned} \tag{8-20}$$

Hence, it yields

$$\gamma_i=\frac{\int\lambda-\frac{1}{i+1}}{\left(\int\lambda-\frac{1}{i+1}\right)(1-u)^{i-1}+\left(\frac{1}{i}-\int\lambda\right)(1-u)^i} \tag{8-21a}$$

$$\gamma_{i+1} = \frac{\frac{1}{i} - \int\lambda}{\left(\int\lambda - \frac{1}{i+1}\right)(1-u)^{i-1} + \left(\frac{1}{i} - \int\lambda\right)(1-u)^{i}} \tag{8-21b}$$

Clearly, γ_i and γ_{i+1} are all nonnegative and let γ_j=0 for $j \notin \{i,\ i+1\}$. Thus, the basic feasible solution to the constraints of the linear program is

$$\gamma(x) = \frac{\int\lambda - \frac{1}{i+1}}{\left(\int\lambda - \frac{1}{i+1}\right)(1-u)^{i-1} + \left(\frac{1}{i} - \int\lambda\right)(1-u)^{i}} x^{i-1}$$

$$+ \frac{\frac{1}{i} - \int\lambda}{\left(\int\lambda - \frac{1}{i+1}\right)(1-u)^{i-1} + \left(\frac{1}{i} - \int\lambda\right)(1-u)^{i}} x^{i} \tag{8-22}$$

(ii) Find a possible feasible solution $\boldsymbol{\beta}$ to the constraints of the dual program

Based on (8-17) the dual program of the primal linear program (8-13) can be expressed as

$$\max\left\{\boldsymbol{\beta}^T\mathbf{b} = \beta_2 \middle| \beta_1\left(\int\lambda - \frac{1}{j}\right) + \beta_2(1-u)^{j-1} \le u^{j-1}, j = 2,3,....\right\} \tag{8-23}$$

where $\boldsymbol{\beta}^T$=[β_1, β_2]. Next, we will specify a possible feasible solution to the constraints of the dual program, which is in relation to the basic feasible solution in (8-22) for the primal program. The two components of the row vector $\mathbf{c}^T$ associated with the basis $\mathbf{B}$ forms the row vector $\mathbf{c}_B^T$=[u^{i-1}, u^i]. According to Theorem 8 in the Appendix J, we define $\boldsymbol{\beta}^T$ for the dual program as

$$\boldsymbol{\beta}^T = \mathbf{c}_B^T \mathbf{B}^{-1}$$

$$= [u^{i-1},\ u^i] \begin{bmatrix} (1-u)^i & -\left(\int \lambda - \frac{1}{i+1}\right) \\ -(1-u)^{i-1} & \int \lambda - \frac{1}{i} \end{bmatrix} \frac{1}{\det(\mathbf{B})} \tag{8-24}$$

Hence, we obtain

$$\beta_1 = \frac{u^i (1-u)^{i-1} - u^{i-1}(1-u)^i}{\left(\int \lambda - \frac{1}{i+1}\right)(1-u)^{i-1} + \left(\frac{1}{i} - \int \lambda\right)(1-u)^i} \tag{8-25a}$$

$$\beta_2 = \frac{\left(\int \lambda - \frac{1}{i+1}\right) u^{i-1} + \left(\frac{1}{i} - \int \lambda\right) u^i}{\left(\int \lambda - \frac{1}{i+1}\right)(1-u)^{i-1} + \left(\frac{1}{i} - \int \lambda\right)(1-u)^i} \tag{8-25b}$$

Obviously, $\boldsymbol{\beta}^T = \mathbf{c}_B^T \mathbf{B}^{-1}$ and $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$ satisfy the requirement that both objective functions have the same value, i.e.,

$$\boldsymbol{\beta}^T \mathbf{b} = \mathbf{c}_B^T \mathbf{x}_B \tag{8-26}$$

(iii) Prove the obtained $\boldsymbol{\beta}$ to be the feasible solution to the constraints of the dual program

Finally, we will verify that the vector $\boldsymbol{\beta}$ fulfills the constraints of the dual program (8-23). More specifically,

$$\beta_1 \left(\int \lambda - \frac{1}{j}\right) + \beta_2 (1-u)^{j-1} \le u^{j-1} \quad \text{for } j = 2, 3, \ldots.. \tag{8-27}$$

such that $\boldsymbol{\beta}$ becomes a feasible solution to the constraints of the dual program. Clearly, it follows from (8-24) that for $j \in \{i,\ i+1\}$ the inequality (8-27) holds exactly with equality as expected. Thus, we only need to prove that for $j \notin \{i,\ i+1\}$ the inequality also holds. The proof can be referred to [60]. Finally, we conclude that $\boldsymbol{\beta}$ is a feasible solution to the constraints of the dual program (8-23).

Based on (8-26) and the Theorem 7 in the Appendix K, the basic feasible solution $\mathbf{x}^T=[0, 0, \ldots, \gamma_i, \gamma_{i+1}, 0, \ldots.]$ is optimal for the primal linear program (8-13). Hence, the optimal solution to the nonlinear program in (8-12) is

$$\lambda_i = \frac{\gamma_i}{\gamma_i + \gamma_{i+1}} = \frac{\int \lambda - \frac{1}{i+1}}{\frac{1}{i} - \frac{1}{i+1}} \tag{8-28a}$$

$$\lambda_{i+1} = \frac{\gamma_{i+1}}{\gamma_i + \gamma_{i+1}} = \frac{\frac{1}{i} - \int \lambda}{\frac{1}{i} - \frac{1}{i+1}} \tag{8-28b}$$

The optimal *left-concentrated* degree distribution $\lambda_c(x)$ is

$$\lambda_c(x) = \frac{\int \lambda - \frac{1}{i+1}}{\frac{1}{i} - \frac{1}{i+1}} x^{i-1} + \frac{\frac{1}{i} - \int \lambda}{\frac{1}{i} - \frac{1}{i+1}} x^{i} \tag{8-29}$$

The minimum value of the objective function of the nonlinear program is

$$\frac{\left(\int \lambda - \frac{1}{i+1}\right) u^{i-1} + \left(\frac{1}{i} - \int \lambda\right) u^{i}}{\left(\int \lambda - \frac{1}{i+1}\right)(1-u)^{i-1} + \left(\frac{1}{i} - \int \lambda\right)(1-u)^{i}} \tag{8-30}$$

Example 8.3: Consider an irregular LDPC codes ensemble whose degree distribution for variable nodes to be optimized is

$$\lambda(x)=\frac{1}{4}x^2+\frac{1}{3}x^3+\frac{1}{4}x^4+\frac{1}{6}x^5 \tag{8-31}$$

We have $\int\lambda=\int_0^1\lambda(x)dx=\frac{11}{45}$. The average degree of a variable node is $i=\lfloor 1/\int\lambda\rfloor=\lfloor 45/11\rfloor=$ 4. Based on (8-28a) and (8-28b) we it yields

$$\lambda_4^{(c)}=\frac{\int\lambda-\frac{1}{i+1}}{\frac{1}{i}-\frac{1}{i+1}}=\frac{\frac{11}{45}-\frac{1}{5}}{\frac{1}{4}-\frac{1}{5}}$$

$$=20\times\frac{11-9}{45}=\frac{8}{9} \tag{8-32a}$$

$$\lambda_5^{(c)}=\frac{\frac{1}{i}-\int\lambda}{\frac{1}{i}-\frac{1}{i+1}}=\frac{\frac{1}{4}-\frac{11}{45}}{\frac{1}{4}-\frac{1}{5}}$$

$$=20\times\frac{45-44}{4\times45}=\frac{1}{9} \tag{8-32b}$$

The optimal *left-concentrated* degree distribution $\lambda_c(x)$ is

$$\lambda_c(x)=\lambda_4^{(c)}x^3+\lambda_5^{(c)}x^4$$

$$=\frac{8}{9}x^3+\frac{1}{9}x^4 \tag{8-33}$$

It is easy to check that $\int\lambda_c=\int_0^1\lambda_c(x)dx=\frac{11}{45}$, which remains the same value as $\int\lambda$ associated with the distribution for variable nodes without concentration. □

8.3 The Optimal Right-Concentrated Degree Distribution

Having studied the optimized degree distribution of variable nodes for an irregular LDPC codes ensemble, i.e., *left-concentrated* degree distribution $\lambda_c(x)$, in this section we will first show that the degree distribution of check nodes can be also optimized by right-concentrated degree distribution given the degree distribution $\rho(x)$ for check nodes, then we present a similar method to determine the optimized distribution by linear program. Finally, we derive the lower bound of the threshold of an irregular LDPC code with the optimized right-concentrated degree distribution, and show that in some cases the performance is improved by increasing the threshold value due to the optimization.

8.3.1 Right-concentrated degree distribution

Recall that the bit error rate of an irregular LDPC code in the lth iteration employing Gallager's decoding algorithm A over BSC is

$$p_l = p_0 - p_0\lambda\left(\frac{1+\rho(1-2p_{l-1})}{2}\right) + (1-p_0)\lambda\left(\frac{1-\rho(1-2p_{l-1})}{2}\right) \tag{8-34}$$

where p_0 is crossover probability of a BSC. For a fixed p_{l-1} and the degree distribution $\lambda(.)$ for variable nodes, p_l can be decreased by increasing $\rho(1-2x)$ through the optimized coefficients ρ_i (i =2, 3,….) over the concerned interval. The optimization procedure is given as follows.

Let **Q** be the set of degree distribution $\rho(x)=\sum_{i\geq 2}\rho_i x^{i-1}$ for check nodes and Q_d be the set of all degree distribution under the constraint $\int\rho = d$. We want to find the concentrated degree distribution ρ_c with $\int\rho = d$ such that

$$\rho_c(1-2x) \geq \rho(1-2x) \text{ for any } x \in \left[0, \frac{d}{2(1+d)}\right] \text{ and } \rho \in Q_d \tag{8-35}$$

Note that $(1-x) \in \left[\frac{1}{d+1}, 1\right]$. Let us investigate the following linear program, which is equivalent to the aforementioned criterion, with the equality and inequality constraints as

$$\min\left\{-\sum_{j\geq 2}\rho_j u^{j-1} \middle| \rho_j \geq 0; \sum_{j\geq 2}\rho_j = 1; \sum_{j\geq 2}\frac{\rho_j}{j} = \int\rho\right\} \tag{8-36}$$

To find a basic feasible solution to the constraints of the primal program, we let $i = \lfloor 1/\int\rho \rfloor$ corresponding to the average degree of variable nodes and thus $u \in \left[\frac{1}{i+1}, 1\right]$ for the primal program. According to the linear program of standard form (8-16), we have the following matrix $\mathbf{A}$ of rank 2.

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & \ldots\ldots & 1 & 1 & \ldots\ldots \\ 1/2 & 1/3 & \ldots\ldots & 1/i & 1/(i+1) & \ldots\ldots \end{bmatrix} \tag{8-37a}$$

The vectors $\mathbf{x}$, $\mathbf{c}$, $\mathbf{b}$ of the primal program are

$$\mathbf{x}^T = [\rho_2,\ \rho_3,\ \ldots\ldots,\ \rho_i,\ \rho_{i+1},\ \ldots\ldots] \tag{8-37b}$$

$$\mathbf{c}^T = [-u,\ -u^2,\ \ldots\ldots,\ -u^{i-1},\ -u^i,\ \ldots\ldots] \tag{8-37c}$$

$$\mathbf{b}^T = [1,\ \textstyle\int\rho] \tag{8-37d}$$

(i) Find a basic feasible solution to the constraints of the primal program

Similar as the left-concentrated degree distribution $\lambda_c(x)$ for variable nodes, let $\mathbf{B}$ be the submatrix consisting of two linearly independent columns of $\mathbf{A}$ as:

$$\mathbf{B}=\begin{bmatrix} 1 & 1 \\ 1/i & 1/(i+1) \end{bmatrix} \tag{8-38}$$

Based on the Theorem 8 in the Appendix J, the two-dimensional row vector $\mathbf{x}_B^T$=[ρ_i, ρ_{i+1}] related to the basis $\mathbf{B}$ is

$$\begin{aligned}\mathbf{x}_B &= \mathbf{B}^{-1}\mathbf{b} \\ &= \begin{bmatrix} 1 & 1 \\ 1/i & 1/(i+1) \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ \int\rho \end{bmatrix} \\ &= \frac{1}{\det(\mathbf{B})} \begin{bmatrix} 1/(i+1) & -1 \\ -1/i & 1 \end{bmatrix} \begin{bmatrix} 1 \\ \int\rho \end{bmatrix} \end{aligned} \tag{8-39}$$

Thus, we have

$$\rho_i = \frac{\int\rho - \frac{1}{i+1}}{\frac{1}{i} - \frac{1}{i+1}} \tag{8-40a}$$

$$\rho_{i+1} = \frac{\frac{1}{i} - \int\rho}{\frac{1}{i} - \frac{1}{i+1}} \tag{8-40b}$$

It is easy to check that $\rho_i \geq 0$ and $\rho_{i+1} \geq 0$ and let $\rho_j = 0$ for $j \notin \{i, i+1\}$. Thus, the basic feasible solution to the constraints of the linear program (8-36) is

$$\rho(x) = \frac{\int\rho - \frac{1}{i+1}}{\frac{1}{i} - \frac{1}{i+1}} x^{i-1} + \frac{\frac{1}{i} - \int\rho}{\frac{1}{i} - \frac{1}{i+1}} x^{i} \tag{8-41}$$

The sub-vector of $\mathbf{c}^T$, associated with $\mathbf{x}_B^T$, is denoted by $\mathbf{c}_B^T$=[$-u^{i-1}$, $-u^i$]. The related value of the objective function is

$$\mathbf{c}_B^T\,\mathbf{x}_B = \frac{\frac{1}{i+1}-\int\rho}{\frac{1}{i}-\frac{1}{i+1}}u^{i-1} + \frac{\int\rho-\frac{1}{i}}{\frac{1}{i}-\frac{1}{i+1}}u^i \tag{8-42}$$

(ii) Find a possible solution $\boldsymbol{\beta}$ to the dual program

The dual program of the primal linear program (8-36) is

$$\text{Max}\left\{\boldsymbol{\beta}^T\mathbf{b} = \beta_1 + \beta_2\int\rho \middle| \beta_1 + \frac{\beta_2}{j} \le -u^{j-1}, j = 2,3,.....\right\} \tag{8-43}$$

where $\boldsymbol{\beta}^T$=[β_1, β_2]. Next we will find a possible feasible solution to the constraints of the dual program, which corresponds to the basic feasible solution to the constraints of the primal program. The two components of the row vector $\mathbf{c}^T$ related to the basis $\mathbf{B}$ forms the row vector $\mathbf{c}_B^T$=[$-u^{i-1}$, $-u^i$]. According to Theorem 8 in the Appendix J, the row vector $\boldsymbol{\beta}^T$ is given as blew for the dual program

$$\begin{aligned}\boldsymbol{\beta}^T &= \mathbf{c}_B^T\mathbf{B}^{-1}\\ &= [-u^{i-1},\ -u^i]\begin{bmatrix}1/(i+1) & -1\\ -1/i & 1\end{bmatrix}\frac{1}{\det(\mathbf{B})}\end{aligned} \tag{8-44}$$

Hence, the components β_1 and β_2 are

$$\beta_1 = \frac{\frac{1}{i+1}u^{i-1} - \frac{1}{i}u^i}{\frac{1}{i}-\frac{1}{i+1}} \tag{8-45a}$$

$$\beta_2 = \frac{u^i - u^{i-1}}{\frac{1}{i} - \frac{1}{i+1}} \tag{8-45b}$$

We can easily check that the objective functions have the same value for both programs, i.e., $\boldsymbol{\beta}^T \mathbf{b} = \mathbf{c}_B^T \mathbf{x}_B$.

(iii) Prove the obtained β to be the feasible solution to the constraints of the dual program

Finally, we will verify that the vector $\boldsymbol{\beta}$ satisfies the inequality constraint $\boldsymbol{\beta}^T \mathbf{A} \le \mathbf{c}^T$ of the dual program (8-43), which is

$$\beta_1 + \frac{\beta_2}{j} \le -u^{j-1} \quad \text{for } j=2, 3, \ldots.. \tag{8-46}$$

such that $\boldsymbol{\beta} = [\beta_1, \beta_2]$ becomes a feasible solution to the dual. Obviously, for $j \in \{i, i+1\}$ the inequality holds exactly with equality as $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$. Thus, we need to further prove that for $j \notin \{i, i+1\}$ the inequality also holds. Applying (8-45a) and (8-45b) to (8-46) and after some manipulations, (8-46) can be rewritten in the form

$$i\left(1 - \frac{i+1}{j}\right) - (i+1)\left(1 - \frac{i}{j}\right)u + u^{j-i} \le 0 \tag{8-47}$$

In general, the left-side expression is a convex function [60] of u, thus we need only verify the validity of the inequality (8-47) at two endpoints, i.e., $u=1$ and $u = \frac{i}{i+1}$. At $u=1$, we have

$$i\left(1-\frac{i+1}{j}\right)-(i+1)\left(1-\frac{i}{j}\right)+1=\frac{i(j-i-1)}{j}-\frac{(i+1)(j-i)}{j}+1$$

$$=\frac{ij-i^2-i-ij-j+i^2+i}{j}+1=0 \tag{8-48}$$

Hence, the inequality holds exactly with equality at $u=1$. At $u=\frac{i}{i+1}$, the left-side of (8-47) is

$$i\left(1-\frac{i+1}{j}\right)-(i+1)\left(1-\frac{i}{j}\right)\frac{i}{i+1}+\left(\frac{i}{i+1}\right)^{j-i}$$

$$=i\frac{j-i-1}{j}-\frac{i(j-i)}{j}+\left(\frac{i}{i+1}\right)^{j-i}=\frac{ij-i^2-i-ij+i^2}{j}+\left(\frac{i}{i+1}\right)^{j-i}$$

$$=-\frac{i}{j}+\left(\frac{i}{i+1}\right)^{j-i} \tag{8-49a}$$

Thus, the inequality (8-47) equivalently becomes

$$i\geq j\left(\frac{i}{i+1}\right)^{j-i} \tag{8-49b}$$

Let us define the following function as

$$w(z)=z\left(\frac{i}{i+1}\right)^{z-i} \tag{8-50a}$$

which is equivalently expressed as

$$w(z)=z\exp\left((z-i)\ln\left(\frac{i}{i+1}\right)\right)\quad \text{for } z>0 \tag{8-50b}$$

The derivative of $w(z)$ with respect to z is

$$\frac{d}{dz}w(z)=\left(\frac{i}{i+1}\right)^{z-i}+z\left(\frac{i}{i+1}\right)^{z-i}\ln\left(\frac{i}{i+1}\right)$$

$$=\left(\frac{i}{i+1}\right)^{z-i}\left(1-z\ln\left(\frac{i+1}{i}\right)\right)$$

$$=\frac{1}{z}\left(\frac{i}{i+1}\right)^{z-i}\left(\frac{1}{z}-\ln\left(\frac{i+1}{i}\right)\right) \tag{8-51}$$

Since $\frac{1}{i+1}<\ln\left(\frac{i+1}{i}\right)<\frac{1}{i}$, thus $w(z)$ is increasing on (0, i] and decreasing on [i+1, $+\infty$). It leads to the following relations

$$w(j)=j\left(\frac{i}{i+1}\right)^{j-i}<w(i)=i\left(\frac{i}{i+1}\right)^{0}=i \quad \text{for } 2\le j<i \tag{8-52a}$$

$$w(j)=j\left(\frac{i}{i+1}\right)^{j-i}<w(i+1)=(i+1)\left(\frac{i}{i+1}\right)=i \quad \text{for } j>i+1 \tag{8-52b}$$

Hence, the inequality (8-47) holds strictly for $j\notin\{i,\ i+1\}$. Finally, we conclude that the inequality (8-47) is valid for any $j\ge2$, which proves the validity of the inequality constraint of the dual program (8-43).

Based on Theorem 7 in the Appendix J the right-concentrated degree distribution obtained in (8-40a) and (8-40b) is optimal. The value of the object function in (8-41) is minimized.

8.3.2 Upper bound of threshold with the optimal right-concentrated degree distribution

In this section we will give the upper bound of the threshold for an irregular LDPC code over a BSC by applying the optimized right-concentrated degree distribution. Meanwhile, we also show that under some condition the performance of an irregular LDPC code can be improved by the increasing threshold.

Let (λ, ρ) be a given degree distribution pair with $\rho \in Q_d$ and with threshold $x_0^*(\lambda,\rho)$. We will show that for the optimized right-concentrated degree distribution ρ_c we have

$$x_0^*(\lambda,\rho_c) \geq \min\left\{x_0^*(\lambda,\rho), \frac{d}{2(1+d)}\right\} \tag{8-53}$$

If $x_0^*(\lambda,\rho)=0$ the above conclusion is proved directly. Hence, assume in the sequel that $x_0^*(\lambda,\rho)>0$. By assumption, for every $0<\varepsilon \leq x_0^*(\lambda,\rho)$ and for all $x \in (0,\ x_0^*(\lambda,\rho)-\varepsilon\,]$

$$(x_0^*(\lambda,\rho)-\varepsilon)-(x_0^*(\lambda,\rho)-\varepsilon)\,\lambda\left(\frac{1+\rho(1-2x)}{2}\right)$$

$$+(1-(x_0^*(\lambda,\rho)-\varepsilon))\,\lambda\left(\frac{1-\rho(1-2x)}{2}\right)<x \tag{8-54}$$

The inequality (8-54) holds since the bit error rate is monotonically decreased with the iterations provided that the crossover probability of BSC is strictly less than the threshold $x_0^*(\lambda,\rho)$. Let

$$z_\varepsilon=\min\left\{x_0^*(\lambda,\rho)-\varepsilon, \frac{c}{2(1+c)}\right\} \tag{8-55}$$

We have known from the previous results that $\rho_c(1-2x) \geq \rho(1-2x)$ for all $x \in [0,\ z_\varepsilon]$. It follows that for $x \in (0,\ z_\varepsilon\,]$

$$z_{\varepsilon}-z_{\varepsilon}\lambda\left(\frac{1+\rho_c(1-2x)}{2}\right)+(1-z_{\varepsilon})\lambda\left(\frac{1-\rho_c(1-2x)}{2}\right)$$

$$\leq z_{\varepsilon}-z_{\varepsilon}\lambda\left(\frac{1+\rho(1-2x)}{2}\right)+(1-z_{\varepsilon})\lambda\left(\frac{1-\rho(1-2x)}{2}\right)$$

$$\leq(x_0^*(\lambda,\rho)-\varepsilon)-(x_0^*(\lambda,\rho)-\varepsilon)\,\lambda\left(\frac{1+\rho(1-2x)}{2}\right)$$

$$+(1-(x_0^*(\lambda,\rho)-\varepsilon))\,\lambda\left(\frac{1-\rho(1-2x)}{2}\right)< x \tag{8-56}$$

The first inequality holds due to the fact that the optimized right-concentrated degree distribution lowers the bit error rate in the current iteration for any given degree distribution λ and the bit error rate x resulted from the last iteration. The second and third inequalities are obvious. This immediately gives the lower bound of the threshold of an irregular LDPC code with optimized right-concentrated degree distribution

$$x_0^*(\lambda,\rho_c)\geq\min\left\{x_0^*(\lambda,\rho)-\varepsilon,\frac{c}{2(1+c)}\right\} \tag{8-57}$$

It easily leads to the inequality (8-53) due to the arbitrary positive number ε fulfilling $0<\varepsilon\leq x_0^*(\lambda,\rho)$. Note that In case of $x_0^*(\lambda,\rho)\leq\frac{c}{2(1+c)}$ we have

$$x_0^*(\lambda,\rho_c)\geq x_0^*(\lambda,\rho) \tag{8-58}$$

which means that the performance of an irregular LDPC code with ρ_c can be enhanced by increasing the threshold in contrast to that with ρ under the same value of $\int\rho$.

8.4 Analyzing the BP Algorithm Using a Gaussian Approximation in AWGN Channel

So far we have focused on the studies on the thresholds of irregular LDPC codes using Gallager's decoding algorithm A over BEC and BSC, and on the design of the optimal degree distribution pair (λ, ρ) for variable and check node of irregular LDPC codes in BSC. In the previous studies we have also investigated the density evolution of regular LDPC codes using Belief Propagation (BP) decoding involving with variable-node and check-node update.

In this section we will present a simple and effective method to estimate the threshold for irregular LDPC codes on a more complicated channel, i.e., memoryless binary-input continuous-output AWGN channel, with Belief Propagation (BP) (or sum-product, message-passing) decoding. This method is based on approximating extrinsic message densities as Gaussian for regular LDPC codes and Gaussian mixtures [16] for irregular LDPC codes. We will show in the sequel that, without much sacrifice in accuracy, a one-dimensional quantity, namely, the mean of a Gaussian density, can act as faithful surrogate for the message density, which is an infinite-dimensional vector.

8.4.1 Introduction of the Symmetry condition

Since a Gaussian is completely specified by its mean and variance, we need to keep the means and variances during iterations. There is an important condition, called the symmetry condition, which is preserved under density evolution for all messages generated from variable and check node to further reduce the estimation of Gaussian distribution in the iterative decoding.

Assume a binary symbol transmitted in an AWGN channel using BPSK modulation scheme, the channel output r_n received at nth instance becomes

$$r_n = s_n + w_n \tag{8-59}$$

where s_n is the nth transmitted binary symbol taking value $+a$ or $-a$ associated with the information bit d_n as 0 and 1 with equal probability, and w_n is an independent Gaussian variable with zero mean and variance σ_0^2 , respectively. Let us evaluate the two conditional probability density functions as follows

$$p(d_n = 0|r_n) = \frac{p(d_n = 0, r_n)}{p(r_n)}$$

$$= \frac{p(r_n|d_n = 0)p(d_n = 0)}{p(r_n|d_n = 0)p(d_n = 0) + p(r_n|d_n = 1)p(d_n = 1)}$$

$$= \frac{p(r_n|d_n = 0)}{p(r_n|d_n = 0) + p(r_n|d_n = 1)} \tag{8-60a}$$

Likewise, we have

$$p(d_n = 1|r_n) = \frac{p(r_n|d_n = 1)}{p(r_n|d_n = 0) + p(r_n|d_n = 1)} \tag{8-60b}$$

The *a posterior* LLR of d_n is

$$L(d_n|r_n) = \log\left(\frac{p(d_n = 0|r_n)}{p(d_n = 1|r_n)}\right) = \log\left(\frac{p(r_n|d_n = 0)}{p(r_n|d_n = 1)}\right)$$

$$=\log\left(\frac{\frac{1}{\sqrt{2\pi}\sigma_0}\exp\left(-\frac{(r_n-a)^2}{2\sigma_0^2}\right)}{\frac{1}{\sqrt{2\pi}\sigma_0}\exp\left(-\frac{(r_n+a)^2}{2\sigma_0^2}\right)}\right)$$

$$=\frac{(r_n+a)^2-(r_n-a)^2}{2\sigma_0^2}=\frac{2a}{\sigma_0^2}r_n \tag{8-61}$$

Obviously, the variable $L(d_n|r_n)$ is distributed by $N\left(\mu_{LLR},\sigma_{LLR}^2\right)$ with $\mu_{LLR}=2a^2/\sigma_0^2$ and σ_L^2 $=4a^2/\sigma_0^2$ conditioned on the binary symbol $s_n=+a$, corresponding to zero information bit, transmitted in the *n*th interval. Hence, it reduces to $\sigma_{LLR}^2=2\mu_{LLR}$. This is the so-called symmetry condition [16][9] for a Gaussian distribution whose mean $\mu>0$ and variance σ^2 fulfill the condition $\sigma^2=2\mu$.

Motivated by the above, we extend this assumption to LLR extrinsic messages generated from variable and check node and approximately distributed by Gaussian law while using BP algorithm in the iterative decoding. Hence, the resulted Gaussian-distributed LLR message is only one dimensional and leads to a great simplification on the subsequent analysis in theory. Note that the same assumption that all $+a$'s symbols (or all-zero information bits) are transmitted is also valid for our derivation.

8.4.2 Gaussian approximation for regular LDPC codes

In this part we will analysis the density evolution of regular LDPC codes by Gaussian approximation. Our complete iteration begins at the variable nodes and then ends at the check nodes, which may slightly differ from the previous analysis regarding the various BP decoding algorithms.

(i) Variable-node update by Gaussian approximation with symmetry condition

For a *regular* LDPC code with variable node of degree d_c in its graph representation and using LLR-BP algorithm, each outgoing LLR extrinsic message $Z^{(l)}$ from a variable node to a related check node in the lth iteration is the sum of the initial *a posterior* LLR F and d_v-1 incoming LLR extrinsic messages $L_1^{(l-1)}$, $L_2^{(l-1)}$, ….., $L_{d_v-1}^{(l-1)}$ from the check nodes in the $(l-1)$th iteration, but one incoming LLR message $L_{d_v}^{(l-1)}$ along the same edge as the outgoing message is excluded. That is

$$Z^{(l)}=F+\sum_{j=1}^{d_v-1}L_j^{(l-1)} \tag{8-62}$$

Assume F and $L_j^{(l-1)}$ ($j=1$, 2, …., d_v-1) be all independent and identical Gaussian variables satisfying the symmetry condition, i.e., $\sigma_F^2=2\mu_F$ and $\left(\sigma_L^{(l-1)}\right)^2=2\mu_L^{(l-1)}$ for all j's. Apparently, $Z^{(l)}$ is also distributed by $N\left(\mu_Z^{(l)},2\mu_Z^{(l)}\right)$ with $\mu_Z^{(l)}$ given by

$$\mu_Z^{(l)}=\mu_F+(d_v-1)\mu_L^{(l-1)} \tag{8-63}$$

We omit the index j in the mean and variance of the variables because the $L_j^{(l-1)}$'s are independent and identically distributed for $1\le j\le d_v-1$. Thus, the Gaussian variable $Z^{(l)}$ can be specified only by its mean value $\mu_Z^{(l)}$. Note that $\mu_L^{(0)}=0$ is set to zero for the first iteration since there is no LLR extrinsic message generated from the associated check nodes.

(ii) Check-node update by Gaussian approximation with symmetry condition

For a *regular* LDPC code with check node of degree d_c in its graph representation and employing LLR-BP algorithm, each outgoing LLR extrinsic message $L^{(l)}$ from a check node to a corresponding variable node in the lth iteration can be generally expressed by a function of d_c-1 incoming LLR extrinsic messages $Z_1^{(l)}$, $Z_2^{(l)}$, ….., $Z_{d_c-1}^{(l)}$ resulted in by the related variable nodes in the same iteration, but one incoming LLR message $Z_{d_c}^{(l)}$ along the same edge as the outgoing message is excluded. That is

$$L^{(l)}=2\tanh^{-1}\left[\prod_{i=1}^{d_c-1}\tanh\left(Z_i^{(l)}/2\right)\right] \tag{8-64a}$$

Equivalently,

$$\tanh\left(L^{(l)}/2\right)=\prod_{i=1}^{d_c-1}\tanh\left(Z_i^{(l)}/2\right) \tag{8-64b}$$

Assume $L^{(l)}$ by (8-64a) be a Gaussian variable fulfilling the symmetry condition. Hence, $L^{(l)}$ is distributed by $N(\mu_L^{(l)},2\mu_L^{(l)})$ and we first need to determine the mean $\mu_Z^{(l)}$ of $Z^{(l)}$ expressed by (8-62) in terms of $L_j^{(l-1)}$ ($j=1, 2, ….,d_v-1$) with mean $\mu_L^{(l-1)}$ obtained in the check-node update at ($l-1$)th iteration.

Let u be a Gaussian variable that fulfills the symmetry condition, i.e., the mean $\mu_u>0$ and variance $\sigma_u^2=2\mu_u$.The expectation $E\left[\tanh(u/2)\right]$ depends only on the mean μ_u of u as

$$E\left[\tanh\left(\frac{u}{2}\right)\right]=\frac{1}{\sqrt{2\pi}\sqrt{2\mu_u}}\int_{-\infty}^{+\infty}\tanh\left(\frac{u}{2}\right)\exp\left(-\frac{(u-\mu_u)^2}{2\times(2\mu_u)}\right)du$$

$$=\frac{1}{\sqrt{4\pi\mu_u}}\int_{-\infty}^{+\infty}\tanh\left(\frac{u}{2}\right)\exp\left(-\frac{(u-\mu_u)^2}{4\mu_u}\right)du \qquad (8\text{-}65)$$

Obviously, $E[\tanh(u/2)]$ is increasing with the mean μ_u since $\tanh(u/2)\in(-1,\ +1)$ is monotonically increasing on $u\in(-\infty,\ +\infty)$. In particular, $E[\tanh(u/2)]=0$ when μ_u approaching to 0, while $E[\tanh(u/2)]=1$ for $\mu_u=+\infty$.

We thus define the following function $\phi(x)$ [16] for $x\in[0,\ +\infty)$, which will be very useful and convenient for further analysis.

$$\phi(x)=\begin{cases}1-\frac{1}{\sqrt{4\pi x}}\int_{-\infty}^{+\infty}\tanh\left(\frac{u}{2}\right)\exp\left(-\frac{(u-x)^2}{4x}\right)du & if \quad x>0\\ 1 & if \quad x=0\end{cases} \qquad (8\text{-}66)$$

It is easy to check that $\phi(x)$ is continuous and monotonically decreasing on $[0,\ +\infty)$ with $\phi(0)=1$ and $\phi(+\infty)=0$. Thus, from (8-63) and (8-64b) we thus obtain the mean $\mu_Z^{(l)}$ by the following steps

$$E\left(\tanh\left(Z_j^{(l)}/2\right)\right)=1-\phi\left(\mu_F+(d_v-1)\mu_L^{(l-1)}\right) \qquad (8\text{-}67a)$$

$$E\left[\prod_{j=1}^{d_c-1}\tanh\left(Z_j^{(l)}/2\right)\right]=\left[E\left(\tanh\left(Z_j^{(l)}/2\right)\right)\right]^{d_c-1} \qquad (8\text{-}67b)$$

$$\mu_L^{(l)}=\phi^{-1}\left(1-E\left[\prod_{j=1}^{d_c-1}\tanh\left(Z_j^{(l)}/2\right)\right]\right)$$

$$=\phi^{-1}\left(1-\left[1-\phi\left(\mu_F+(d_v-1)\mu_L^{(l-1)}\right)\right]^{d_c-1}\right) \tag{8-67c}$$

where $\mu_L^{(l-1)}=0$ is set to zero for $l=1$ since no LLR extrinsic message is sent from the associated check nodes.

(iii) Upper and lower bounds on $\phi(x)$ for $x>0$

For $x>0$ the function $\phi(x)$ is

$$\phi(x)=1-\frac{1}{\sqrt{4\pi x}}\int_{-\infty}^{+\infty}\frac{\exp(u)-1}{\exp(u)+1}\exp\left[-\frac{(u-x)^2}{4x}\right]du$$

$$=1-\frac{1}{\sqrt{4\pi x}}\int_{-\infty}^{+\infty}\left(1-\frac{2}{\exp(u)+1}\right)\exp\left[-\frac{(u-x)^2}{4x}\right]du$$

$$=\frac{1}{\sqrt{\pi x}}\int_{-\infty}^{+\infty}\frac{1}{1+\exp(u)}\exp\left(-\frac{(u-x)^2}{4x}\right)du \tag{8-68}$$

If $u<0$ we have

$$\frac{1}{1+\exp(u)}=\frac{1}{1-[-\exp(u)]}$$

$$=1-e^{u}+e^{2u}-e^{3u}+e^{4u}+\ldots\ldots$$

$$=\sum_{k=0}^{+\infty}(-1)^k\exp(ku) \tag{8-69a}$$

Likewise, we have for $u>0$

$$\frac{1}{1+\exp(u)}=\frac{\exp(-u)}{1-[-\exp(-u)]}$$

$$=e^{-u}-e^{-2u}+e^{-3u}-e^{-5u}+e^{-6u}$$

$$=\sum_{k=0}^{+\infty}(-1)^k \exp[-(k+1)u] \tag{8-69b}$$

Based on (8-68) and (8-69) $\phi(x)$ can be further expressed as

$$\phi(x)=\frac{1}{\sqrt{\pi x}}\int_{-\infty}^{0}\sum_{k=0}^{+\infty}(-1)^k \exp(ku)\exp\left(-\frac{(u-x)^2}{4x}\right)du$$

$$+\frac{1}{\sqrt{\pi x}}\int_{0}^{+\infty}\sum_{k=0}^{+\infty}(-1)^k \exp[-(k+1)u]\exp\left(-\frac{(u-x)^2}{4x}\right)du$$

$$=\frac{2}{\sqrt{\pi x}}\exp\left(-\frac{x}{4}\right)\int_{0}^{+\infty}\sum_{k=0}^{+\infty}(-1)^k \exp\left[-\left(k+\frac{1}{2}\right)u\right]\exp\left[-\frac{u^2}{4x}\right]du$$

$$=\frac{2}{\sqrt{\pi x}}\exp\left(-\frac{x}{4}\right)\exp\left(\frac{(2k+1)^2 x}{4}\right)\int_{0}^{+\infty}\sum_{k=0}^{+\infty}(-1)^k \exp\left[-\frac{\left(u+(2k+1)x\right)^2}{4x}\right]du$$

$$=4\sum_{k=0}^{+\infty}(-1)^k \exp\left[(k^2+k)x\right]\left\{\frac{1}{\sqrt{2\pi}\sqrt{2x}}\int_{0}^{+\infty}\exp\left[-\frac{\left(u+(2k+1)x\right)^2}{2\times 2x}\right]du\right\}$$

$$=4\sum_{k=0}^{+\infty}(-1)^k \exp\left[(k^2+k)x\right]Q\left(\sqrt{\frac{x}{2}}(2k+1)\right) \tag{8-70}$$

where

$$Q(y)=\frac{1}{\sqrt{2\pi}}\int_{y}^{+\infty}\exp\left(-\frac{t^2}{2}\right)dt \quad \text{for } y>0 \tag{8-71}$$

is the well-known Q-function. Further applying

$$\left(\frac{1}{x}-\frac{1}{x^3}\right)\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right)<Q(x)<\left(\frac{1}{x}\right)\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right) \tag{8-72}$$

[67], we get

$$\phi(x) < 4\sum_{k=0}^{+\infty}(-1)^k \exp\left[(k^2+k)x\right]\frac{1}{\sqrt{\frac{x}{2}}(2k+1)}\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(2k+1)^2 x}{4}\right)$$

$$=\frac{4}{\sqrt{\pi x}}\exp\left(-\frac{x}{4}\right)\sum_{k=0}^{+\infty}\frac{(-1)^k}{2k+1} < \sqrt{\frac{\pi}{x}}\exp\left(-\frac{x}{4}\right) \quad (8\text{-}73)$$

where we use the equality [67][68]

$$\sum_{k=0}^{+\infty}\frac{(-1)^k}{2k+1}=\frac{\pi}{4} \quad (8\text{-}74)$$

Similarly, we get

$$\phi(x) > 4\sum_{k=0}^{+\infty}(-1)^k \exp\left[(k^2+k)x\right]\left(\frac{1}{\sqrt{\frac{x}{2}}(2k+1)}-\frac{1}{\sqrt{\frac{x}{2}}(2k+1)}\frac{1}{\frac{x}{2}(2k+1)^2}\right)\times$$

$$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(2k+1)^2 x}{4}\right)=\frac{4}{\sqrt{\pi x}}\exp\left(-\frac{x}{4}\right)\sum_{k=0}^{+\infty}(-1)^k\left(\frac{1}{2k+1}-\frac{2}{x(2k+1)^3}\right)$$

$$=\frac{4}{\sqrt{\pi x}}\exp\left(-\frac{x}{4}\right)\sum_{k=0}^{+\infty}\left(\frac{(-1)^k}{2k+1}+\frac{2\times(-1)^{k+1}}{x(2k+1)^3}\right) \quad (8\text{-}75a)$$

The second term of right-side of (8-75a) is negative if k is an even number. Finally, we establish the inequality as

$$\phi(x) > \frac{4}{\sqrt{\pi x}}\exp\left(-\frac{x}{4}\right)\left[\sum_{k=0}^{+\infty}\frac{(-1)^k}{2k+1}-\sum_{j=0}^{+\infty}\frac{2}{x(4j+1)^3}\right]$$

$$> \frac{4}{\sqrt{\pi x}}\exp\left(-\frac{x}{4}\right)\left(\frac{\pi}{4}-\frac{2}{x}\times\frac{3\pi}{8}\right)=\sqrt{\frac{\pi}{x}}\exp\left(-\frac{x}{4}\right)\left(1-\frac{3}{x}\right) \quad (8\text{-}75b)$$

where we use

$$\sum_{j=0}^{+\infty}\frac{1}{(4j+1)^3} \tag{8-75c}$$

Finally, it results in

$$\sqrt{\frac{\pi}{x}}\exp\left(-\frac{x}{4}\right)\left(1-\frac{3}{x}\right)<\phi(x)<\sqrt{\frac{\pi}{x}}\exp\left(-\frac{x}{4}\right)\ \text{ for } x>0 \tag{8-76}$$

(iv) The exact and approximate thresholds for various regular LDPC codes

The aforementioned bounds, which are tight as x approaching to infinity, will be used in analyzing the behavior of the BP-based decoder and in optimizing degree distributions in AWGN channel. We know from (8-66) and (8-67) that $\mu_L^{(l)}$, which is approaching to infinity with increasing number of decoding iterations l, implies that a BP-based decoder asymptotically achieves error-free transmission over a noisy channel.

Although it is possible to calculate the thresholds and to optimize degree distributions using the exact values of $\phi(x)$, two approximations [16] can be used instead to speed up the calculations without much sacrifice in accuracy. For small x, say x<10, we can use a good approximation $\phi(x)\approx\exp\left(\alpha x^{\gamma}+\beta\right)$ instead of (8-76) for this range of x, where α= –0.4527, β=0.0218 and γ=0.86. For large x, say x>10, we use the average of the upper and lower bounds of (8-76) as an approximation for $\phi(x)$.

Table 8.1 presents the approximate thresholds σ_{GA} calculated using (8-67c) compared to the exact thresholds σ_{exact} from density evolution. There is about 0.03dB to 0.1dB gap ($20\log_{10}\left(\sigma_{exact}/\sigma_{GA}\right)$dB) compared to the exact values.

d_v	d_c	rate	σ_{GA}	σ_{exact}	Error [dB]
3	6	0.5	0.8747	0.8809	0.06
4	8	0.5	0.8323	0.8376	0.06
5	10	0.5	0.7910	0.7936	0.03
3	5	0.4	1.0003	1.0093	0.08
4	6	1/3	1.0035	1.0109	0.06
3	4	0.25	1.2517	1.2667	0.10
4	10	0.6	0.7440	0.7481	0.05
3	9	2/3	0.7051	0.7082	0.04
3	12	0.75	0.6297	0.6320	0.03

Table 8.1 Approximate and exact threshold values for various (d_v, d_c) regular LDPC codes for the binary-input continuous-output AWGN channel and BP decoding algorithm. [Ref.16, Table I]

Fig. 8.1 shows differences between approximate and exact thresholds for various regular LDPC codes, where SNR_{norm} is defined as the distance from the Shannon limit σ_{opt} in decibels, i.e., $SNR_{norm}=10\log_{10}\left(\sigma_{opt}^2/\sigma_T^2\right)$ using σ_T as σ_{GA} and σ_{exact} given by Table 8.1 for the respective curves. The Shannon limit is σ_{opt}=0.979 for all rate-1/2 error correcting codes. Thus, for a (3, 6) regular LDPC code the distance between the approximate threshold and the Shannon limit is $20\log_{10}(0.979/0.8747)$=0.9785, while the distance is 20 $\log_{10}(0.979/0.8809)$=0.9171 with respect to the exact threshold.

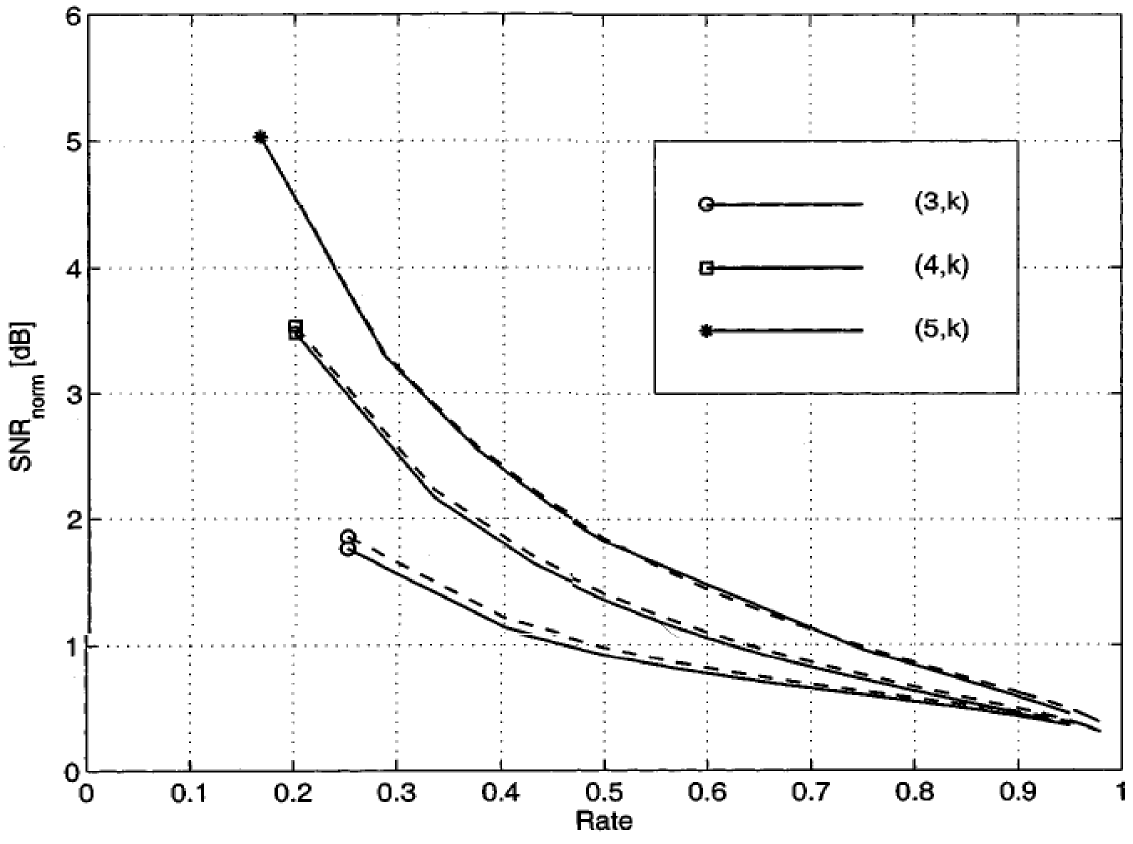


Fig. 8.1 Approximate thresholds, denoted by dash lines, and exact thresholds, denoted by solid straight lines, for various (d_v, d_c) regular LDPC codes with d_v=3, 4 and 5, respectively.

It is very interesting to note that for the same rate the accuracy becomes improved as d_v becomes large, which can be explained because as more inputs are added at check and variable nodes, the output becomes more Gaussian as we assume. This fact is also demonstrated by the regular LDPC codes of same rate in Table 8.1. The error in dB between σ_{GA} and σ_{exact} for a (3, 6) regular LDPC code is 0.06, while the error is greatly reduced to 0.03dB for a (5, 10) regular LDPC code.

8.4.3 Gaussian approximation for irregular LDPC codes starting from check-node update

In this part we will naturally extend the preceding analysis to irregular LDPC codes over an AWGN channel. We again consider an ensemble of random irregular LDPC codes with degree distributions $\lambda(x)=\sum_{j=2}^{d_v}\lambda_j x^{j-1}$ and $\rho(x)=\sum_{j=2}^{d_c}\rho_j x^{j-1}$ for variable and check node, respectively. Before commencing the analysis of irregular LDPC codes by Gaussian approximation, besides all-zero information bits (or all $+a$'s binary symbols) transmitted over an AWGN channel, the two additional assumptions are made by empirical evidence [16] as follows:

(1) An individual output of a variable or a check node with a definite degree distribution is assumed to be a Gaussian density that satisfies the symmetry condition introduced in the sequel.

(2) A node will get independent and identically distributed LLR messages from its neighbors, where each of these messages is a Gaussian mixture [16] of different Gaussian densities from neighbors with different degrees.

This first hypothesis is completely the same as in the preceding analysis for regular LDPC codes based on Gaussian approximation. In the second hypothesis the component Gaussian densities of a Gaussian mixture is based the first one, which fulfill the symmetry condition.

(i) Variable-node update by Gaussian approximation and Gaussian mixture

Similar as (8-62) for regular LDPC codes, the mean of an outgoing LLR extrinsic message $Z_i^{(l)}$ from a variable node of a degree$-i$ (i=2, 3, …, d_v) to an associated check node at the lth iteration is given by

$$\mu_{Z,i}^{(l)}=\mu_F+(i-1)\,\mu_L^{(l-1)} \tag{8-77}$$

where μ_F is the mean of the Gaussian variable F, $\mu_L^{(l-1)}$ is the means of an outgoing LLR extrinsic message $L^{(l-1)}$ from a related check node at the ($l-1$)th iteration. Note that based on the second assumption $L^{(l-1)}$ is not Gaussian but a Gaussian mixture made up of various different Gaussian densities, each of which fulfills the symmetry condition based on the first assumption.

Similarly, according to the first assumption $Z_i^{(l)}$ is a Gaussian variable satisfying the symmetry condition and distributed by $N(\mu_{Z,i}^{(l)},2\mu_{Z,i}^{(l)})$ ($\mu_{Z,i}^{(l)}$ >0). Thus, statistically an incoming message $Z^{(l)}$ to an associated check node at the lth iteration will have the following Gaussian mixture probability density function (pdf) as

$$f_Z^{(l)}=\sum_{i=2}^{d_v}\lambda_i N\left(\mu_{Z,i}^{(l)},2\mu_{Z,i}^{(l)}\right)\quad(\mu_{Z,i}^{(l)}>0) \tag{8-78}$$

It is very important to note that a Gaussian mixture is not a Gaussian distribution, but a Gaussian-like one. The next example will provide a better understanding for the so-called Gaussian mixture distribution.

Example 8.4: Assume a binary symbol transmitted over an AWGN channel using BPSK modulation scheme, the channel output r_n received at nth instance is $r_n=s_n+w_n$, where s_n

is the *n*th transmitted symbol taking value $+a$ or $-a$ related to the information bit d_n as 0 or 1 with equal probability, and w_n is a Gaussian noise variable with zero mean and variance σ_0^2, respectively.

The unconditional probability density function of r_n is calculated by taking the two possible binary transmitted symbols $+a$ and $-a$ with their respective probabilities into account

$$p_r(r_n) = \frac{1}{2}\frac{1}{\sqrt{2\pi}\sigma_0}\exp\left(-\frac{(r_n-a)^2}{2\sigma_0^2}\right) + \frac{1}{2}\frac{1}{\sqrt{2\pi}\sigma_0}\exp\left(-\frac{(r_n+a)^2}{2\sigma_0^2}\right)$$

$$= \frac{1}{2}N\left(+a,\sigma_0^2\right) + \frac{1}{2}N\left(-a,\sigma_0^2\right) \tag{8-79a}$$

The above density is a Gaussian mixture distribution with equal probability for each conditional Gaussian distribution.

Based on (8-61) the *a posterior* LLR of d_n, denoted by ω_n for $L(d_n|r_n)$, is also distributed by a Gaussian mixture density as

$$p_\omega(\omega_n) = \frac{1}{2}\frac{1}{\sqrt{2\pi}\sigma_w}\exp\left(-\frac{(\omega_n-\mu_\omega)^2}{2\sigma_w^2}\right) + \frac{1}{2}\frac{1}{\sqrt{2\pi}\sigma_w}\exp\left(-\frac{(\omega_n+\mu_\omega)^2}{2\sigma_w^2}\right)$$

$$= \frac{1}{2}N\left(+\mu_\omega,\sigma_\omega^2\right) + \frac{1}{2}N\left(-\mu_\omega,\sigma_\omega^2\right) \tag{8-79b}$$

where $\mu_w = 2a^2/\sigma_0^2$, corresponding to the transmitted symbol $+a$ (or information bit $d_n=$ 0), and $\sigma_w^2 = 2\mu_w$. Each of the conditional Gaussian distribution fulfills the symmetry condition. The only minor difference between the example and the Gaussian mixture for the irregular LDPC codes is that the mean is always assumed to be nonnegative for the latter. □

(ii) Check-node update by Gaussian approximation and Gaussian mixture

According to the Gaussian mixture in (8-78) we get

$$E\left(\tanh\left(\frac{Z^{(l)}}{2}\right)\right)=\sum_{i=2}^{d_v}\lambda_i\frac{1}{\sqrt{4\pi\mu_{Z,i}^{(l)}}}\int_{-\infty}^{+\infty}\tanh\left(\frac{z^{(l)}}{2}\right)\exp\left(-\frac{\left(z^{(l)}-\mu_{Z,i}^{(l)}\right)^2}{4\mu_{Z,i}^{(l)}}\right)dz^{(l)}$$

$$=1-\sum_{i=2}^{d_v}\lambda_i\left(1-\frac{1}{\sqrt{4\pi\mu_{Z,i}^{(l)}}}\int_{-\infty}^{+\infty}\tanh\left(\frac{z^{(l)}}{2}\right)\exp\left(-\frac{\left(z^{(l)}-\mu_{Z,i}^{(l)}\right)^2}{4\mu_{Z,i}^{(l)}}\right)dz^{(l)}\right)$$

$$=1-\sum_{i=2}^{d_v}\lambda_i\phi(\mu_{Z,i}^{(l)}) \tag{8-80}$$

Then, we will calculate the mean of an output LLR extrinsic message $L^{(l)}$ of a check node at the lth iteration. First, let $\mu_{L,j}^{(l)}$ be the mean of the output message $L_j^{(l)}$ of a check node with degree j, which is evaluated by similar way as (8-64b) for regular LDPC codes.

$$\mu_{L,j}^{(l)}=\phi^{-1}\left(1-\left[E\left(\tanh\left(\frac{Z^{(l)}}{2}\right)\right)\right]^{j-1}\right)$$

$$=\phi^{-1}\left(1-\left[1-\sum_{i=2}^{d_v}\lambda_i\phi\left(\mu_{Z,i}^{(l)}\right)\right]^{j-1}\right) \tag{8-81}$$

Note that $L^{(l)}$ is a Gaussian mixture conditioned on $L^{(l)}=L_j^{(l)}$ with probability ρ_j for $2\leq j\leq d_c$, while $L_j^{(l)}$ is distributed by Gaussian density with $(\sigma_{L,j}^{(l)})^2=2\mu_{L,j}^{(l)}$ ($\mu_{L,j}^{(l)}>0$). Therefore, by linearly combing these means for check nodes of degree-j with weight ρ_j, we get the mean of $L^{(l)}$ as given in the following:

$$\mu_L^{(l)}=\sum_{j=2}^{d_c}\rho_j\mu_{L,j}^{(l)}=\sum_{j=2}^{d_c}\rho_j\phi^{-1}\left(1-\left[1-\sum_{i=2}^{d_v}\lambda_i\phi\left(\mu_{Z,i}^{(l)}\right)\right]^{j-1}\right)$$

$$=\sum_{j=2}^{d_c}\rho_j\phi^{-1}\left(1-\left[1-\sum_{i=2}^{d_v}\lambda_i\phi\left(\mu_F+(i-1)\mu_L^{(l-1)}\right)\right]^{j-1}\right) \tag{8-82}$$

The above mean value $\mu_L^{(l)}$ will be used to calculate the mean $\mu_{Z,i}^{(l+1)}$ by using (8-77) in the (l+1)th iteration.

For $0<s<+\infty$ and $0\leq t<+\infty$, we define two functions as

$$f_j\left(s,t\right)=\phi^{-1}\left(1-\left[1-\sum_{i=2}^{d_v}\lambda_i\phi\left(s+(i-1)t\right)\right]^{j-1}\right) \tag{8-83a}$$

$$f\left(s,t\right)=\sum_{j=2}^{d_c}\rho_j f_j(s,t) \tag{8-83b}$$

Therefore, we rewrite (8-82) by the recursive expression as

$$t_l=f(s,t_{l-1}) \tag{8-84}$$

where $s=\mu_F$ and $t_l=\mu_L^{(l)}$. The initial value t_0 is 0. Since $t_1=f(s,0)>0$ for $s>0$, the iteration (8-84) will always start. Note that the recursive formula (8-84) will be very helpful for optimizing the check node degree $\rho(x)$ for an irregular LDPC code by liner program since (8-83b) is linear in the ρ_i's.

(iii) Convergence condition for BP decoding algorithm using Gaussian approximation over an AWGN channel

Recall that for Gallager's algorithm A in BSC where the bit error probability converges to zero provided that the crossover probability p is less than the threshold ε^*, for BP decoding algorithm using Gaussian approximation over AWGN channel there also exists a threshold in term of $s^*=\mu_F^*$ that is the infimum of all s (or μ_F) in $(0, +\infty)$ such that $t_l(s)$ converges to $+\infty$ as $l\rightarrow+\infty$.

Since $\varphi(x)$ is monotonically increasing on both $0<s<+\infty$ and $0\leq t<+\infty$. By inducting, we conclude that for all $s>s^*$ and $t_l(s)>t_l(s^*)$. The following theorem gives a sufficient and necessary condition for the convergence of $t_l(s)$ as l approaches to $+\infty$.

Theorem 8.2: $t_l(s)$ will converge to $+\infty$ if and only if $t<f(s,t)$ for all $t\in(0, +\infty)$.

The proof of the theorem can be referred to the Appendix K.

8.4.4 Gaussian approximation for irregular LDPC codes starting from variable-node update

(i) An alternative expression for iterative decoding using Gaussian approximation

Alternatively, we can formulate (8-84) by a similar method starting from the prospect of check-node update, which is based on the following steps:

Step 1. Based on (8-80) we calculate the following equation with respect to the $(l-1)$th iteration

$$E\left(\tanh\left(\frac{Z^{(l-1)}}{2}\right)\right)=1-\sum_{i=2}^{d_v}\lambda_i\phi(\mu_{Z,i}^{(l-1)})$$

$$=1-r_{l-1} \qquad (8\text{-}85)$$

where $r_{l-1}=\sum_{i=2}^{d_v}\lambda_i\phi\left(\mu_{Z,i}^{(l-1)}\right)$ and clearly $0\le r_{l-1}\le 1$ since $\phi\left(\mu_{Z,i}^{(l-1)}\right)\in[0,\ 1]$ for each $2\le i\le d_v$.

Step 2. We will calculate the mean $\mu_{L,j}^{(l)}$ of the Gaussian variable $L_j^{(l)}$ generated from a check node of degree-j at the l-th iteration. According to (8-64b) for check-node update, it becomes

$$\mu_{L,j}^{(l)}=\phi^{-1}\left[1-\left(E\left(\tanh\left(\frac{Z^{(l-1)}}{2}\right)\right)\right)^{j-1}\right]$$

$$=\phi^{-1}\left(1-(1-r_{l-1})^{j-1}\right) \qquad (8\text{-}86)$$

Step 3. The mean $\mu_L^{(l)}$ of variable $L^{(l)}$ distributed by a Gaussian mixture density at the l-th iteration is the linear combination of $\mu_{L,j}^{(l)}$ for j=2, 3,…, d_c, i.e.,

$$\mu_L^{(l)}=\sum_{j=2}^{d_c}\rho_j\mu_{L,j}^{(l)}=\sum_{j=2}^{d_c}\rho_j\phi^{-1}\left(1-(1-r_{l-1})^{j-1}\right) \qquad (8\text{-}87)$$

Step 4. The mean $\mu_{Z,i}^{(l)}$ of the Gaussian variable $Z_i^{(l)}$, generated from a variable of degree-i (i=2, 3, …., d_v) at the l-th iteration, is evaluated by $\mu_{Z,i}^{(l)}=\mu_F+(i-1)\mu_L^{(l)}$ based on (8-77) for variable-node update.

Step 5. Finally, we have the following equation at the lth iteration corresponding to that at the ($l-1$)th iteration as

$$E\left(\tanh\left(\frac{Z^{(l)}}{2}\right)\right)=1-r_l \tag{8-88a}$$

where r_l is

$$\begin{aligned} r_l &= \sum_{i=2}^{d_v} \lambda_i \phi(\mu_{Z,i}^{(l)}) = \sum_{i=2}^{d_v} \lambda_i \phi\left(\mu_F + (i-1)\mu_L^{(l)}\right) \\ &= \sum_{i=2}^{d_v} \lambda_i \phi\left(\mu_F + (i-1)\sum_{j=2}^{d_c} \rho_j \mu_{L,j}^{(l)}\right) \\ &= \sum_{i=2}^{d_v} \lambda_i \phi\left(\mu_F + (i-1)\sum_{j=2}^{d_c} \rho_j \phi^{-1}\left(1-\left(1-r_{l-1}\right)^{j-1}\right)\right) \end{aligned} \tag{8-88b}$$

Let $0<s<+\infty$ and $0<r\le 1$, we define two functions $h_i(s,r)$ and $h(s,r)$ as

$$h_i(s,r)=\phi\left(s+(i-1)\sum_{j=2}^{d_c}\rho_j\phi^{-1}\left(1-(1-r)^{j-1}\right)\right) \tag{8-89a}$$

$$h(s,r)=\sum_{i=2}^{d_v}\lambda_i h_i(s,r) \tag{8-89b}$$

Hence, the recursive expression (8-88b) becomes

$$r_l = h(s, r_{l-1}) \tag{8-90}$$

where $s=\mu_F$ and the initial value $r_0=\sum_{i=2}^{d_v}\lambda_i\phi(\mu_F)=\phi(s)$. Note that the expression (8-90) will be useful in optimizing the variable node degree $\lambda(x)$ by using liner program since (8-89b) is linear in the λ_i's.

(ii) Convergence condition in terms of r_l and the relationship between t_l

According to the definitions for t_l and r_l

$$t_l = \mu_L^{(l)} \tag{8-91a}$$

$$r_l = \sum_{i=2}^{d_v} \lambda_i \phi\left(\mu_{Z,i}^{(l)}\right) = \sum_{i=2}^{d_v} \lambda_i \phi\left(\mu_F + (i-1)\mu_L^{(l)}\right) \tag{8-91b}$$

It is easy to get that $r_l \to 0$ if and only if $t_l \to +\infty$. Clearly, $h(s,r)$ is also an increasing function on $0 < r \le 1$. Similar as the Theorem 8.2, we can easily show that the convergence condition $r_l(s) \to 0$ is fulfilled if and only if

$$r > h(s,r) \quad \forall\ r \in (0,\ \phi(s)) \tag{8-92}$$

8.4.5 Simulation results of approximate and exact densities at the input of check and variable nodes

In order to show how well message densities are approximated by Gaussian mixture, let us consider the following rate-1/2 irregular LDPC code with degree distributions [16][69]

$$\lambda(x) = 0.23403\,x + 0.21242\,x^2 + 0.14690\,x^5 + 0.10284\,x^6 + 0.30381\,x^{19} \tag{8-93a}$$

$$\rho(x) = 0.71875\,x^7 + 0.28125\,x^8 \tag{8-93b}$$

The threshold is 0.9669 and 0.9473 [69] that are calculated by density evolution and Gaussian approximation, respectively. In the simulation the noise power is set to be 0.006dB [16] below the threshold in each case.

Fig. 8.2 shows two message densities from variable to check nodes when density evolution and the Gaussian approximation are used for the code in (8-93). The both results are very close [16].

Fig. 8.3 shows corresponding densities from check to variable nodes. In this case, the exact density has a spike at 0, which cannot be modeled by a Gaussian approximation [16]. The fact that the two densities in the other direction are very close suggests that the approximation is working well. Actually, even though the exact density in Fig. 8.3 is different from a Gaussian, the output of the variable node will be Gaussian-like again because many independent identically distributed inputs are added at the variable node.

Such a mismatch between the exact and approximate densities in Fig. 8.3 may be due to the inherit nature of check node computations (the tanh rule), not because the check node distribution $\rho(x)$ is irregular [16].

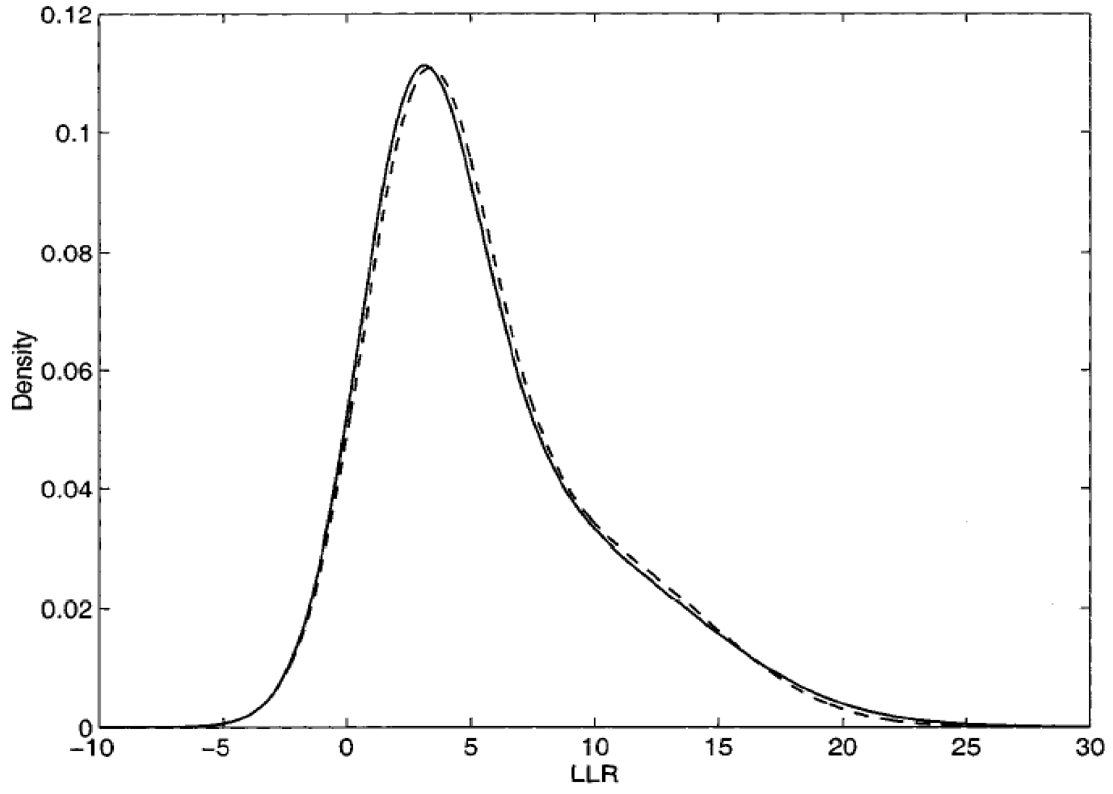


Fig. 8.2 Approximate (---) and (–) densities at the check node input.

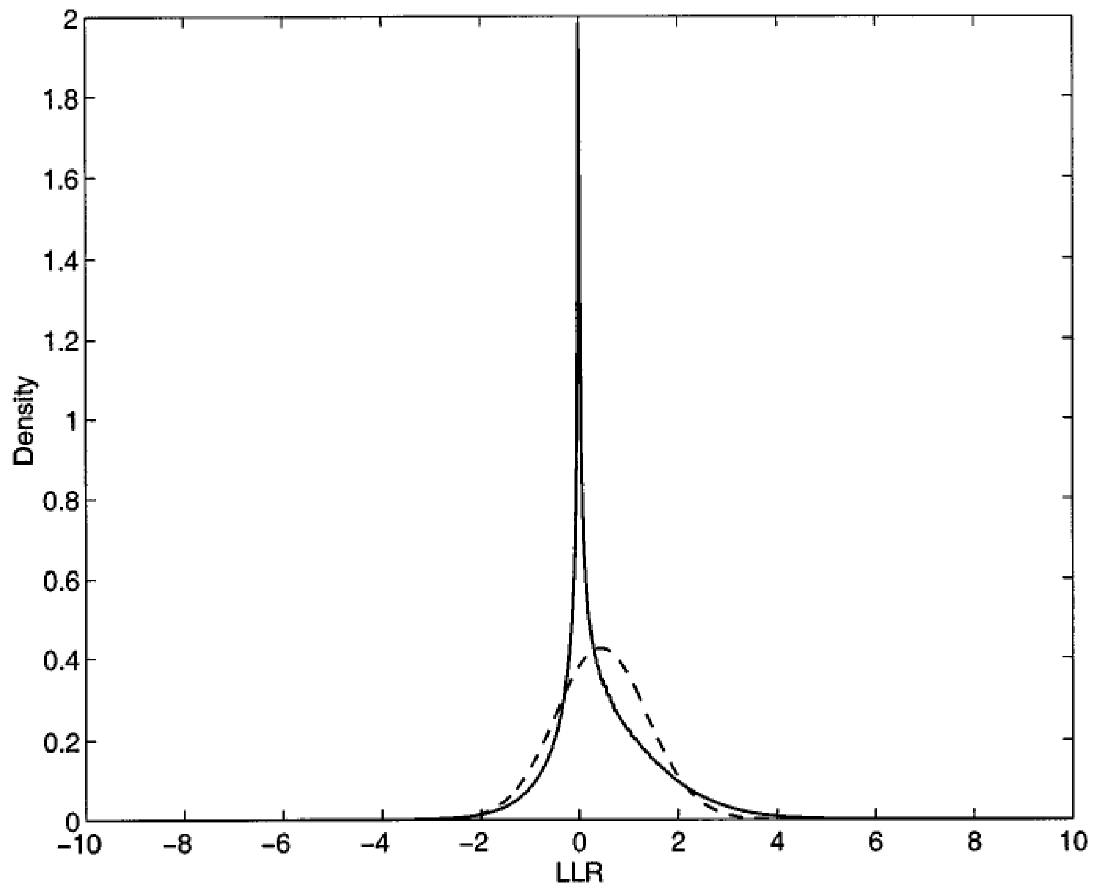


Fig. 8.3 Approximate (---) and (–) densities at the variable node input.

8.5 Stability Issues and Asymptotic Bit Error Rate under the Gaussian Approximation over an AWGN Channel

In this section, we will find the condition for the convergence of t_l to $+\infty$ in (8-84) as l tends to infinity when t_0 is large enough, which is know as the stability condition. We also derive the rate of the convergence of the probability of error using this result.

8.5.1 The stability condition under the Gaussian approximation

We first prove the following theorem that gives the behavior of (8-83a) and (8-83b) when t is large

Theorem 8.3: As $t \to +\infty$, $\triangle t = f(s,t) - t$ becomes

$$\triangle t = s + (i_1 - 2)t - 4\log\lambda_{i_1} - 4\sum_{j=2}^{d_c}\left[\rho_j \log(j-1)\right] + O(t^{-1}) \tag{8-94}$$

where λ_{i_1} is the first nonzero λ_i and $s = 2a^2/\sigma_0^2$. □

The proof of the theorem can be referred to the Appendix K.

If i_1>2, then $\triangle t$ is always greater than zero regardless of s and $\rho(x)$ as t approaches to infinity. Otherwise, i.e., i_1=2, $\triangle t$ becomes a constant as t becomes large. In this case we let a=1 that results in s=2/σ_0^2, $\triangle t$ is expressed as

$$\begin{aligned}\triangle t &= 4\log\left(\exp\left(\frac{1}{2\sigma_0^2}\right)\right) - 4\log\lambda_2 - 4\log\left[\prod_{j=2}^{d_c}(j-1)^{\rho_j}\right] + O(t^{-1}) \\ &= 4\log\left(\frac{\exp\left(1/\left(2\sigma_0^2\right)\right)}{\prod_{j=2}^{d_c}(j-1)^{\rho_j}}\right) - 4\log\lambda_2 + O(t^{-1}) \\ &= 4\log\left(\frac{\lambda_2^*}{\lambda_2}\right) + O(t^{-1}) \end{aligned} \tag{8-95}$$

where λ_2^* is

$$\lambda_2^* = \frac{\exp\left(1/\left(2\sigma_0^2\right)\right)}{\prod_{j=2}^{d_c}(j-1)^{\rho_j}} \tag{8-96}$$

Therefore, if $\lambda_2 < \lambda_2^*$, then Δt>0 when t is large enough, and t_l will converge to infinity as l trends to infinity with t_0 large enough. If $\lambda_2 > \lambda_2^*$, then Δt<0 when t is large enough, and t_l cannot converge to infinity regardless of its initial value. Note that i_1>2 implies that λ_2 =0 that clearly satisfies the condition $\lambda_2 < \lambda_2^*$, and we have $t_l \to +\infty$ as l trends to infinity.

Recall that for an irregular LDPC code with a degree distribution pair ($\lambda(x)$, $\rho(x)$) using BP decoding algorithm, the stability condition [8] over binary-input continuous-output AWGN channel is

$$\lambda'(0)\rho'(1) < \exp\left(\frac{1}{2\sigma_0^2}\right) \tag{8-97a}$$

Equivalently,

$$\begin{aligned}\lambda'(0) = \lambda_2 &< \frac{\exp\left(1/\left(2\sigma_0^2\right)\right)}{\rho'(1)} \\ &= \frac{\exp\left(1/\left(2\sigma_0^2\right)\right)}{\sum_{j=2}^{d_c}\left(\rho_j(j-1)\right)}\end{aligned} \tag{8-97b}$$

Using Jensen's inequality [8], we conclude that

$$\prod_{j=2}^{d_c}(j-1)^{\rho_j} \le \sum_{j=2}^{d_c}\left(\rho_j(j-1)\right) \tag{8-98}$$

for all $\rho(x)$, with equality if and only if ρ_k=1 for some $k \ge 2$. This implies that

$$\lambda_2^* > \frac{\exp\left(1/\left(2\sigma_0^2\right)\right)}{\sum_{j=2}^{d_c}\left(\rho_j(j-1)\right)} \tag{8-99}$$

The above inequality means that the stable degree distributions under the Gaussian approximation may be stable under density evolution. For example, if we choose λ_2 as

$$\frac{\exp\left(1/\left(2\sigma_0^2\right)\right)}{\sum_{j=2}^{d_c}\left(\rho_j\,(j-1)\right)} < \lambda_2 < \lambda_2^* = \frac{\exp\left(1/\left(2\sigma_0^2\right)\right)}{\prod_{j=2}^{d_c}(j-1)^{\rho_j}} \tag{8-100}$$

Then, in this scene the distribution pair ($\lambda(x)$, $\rho(x)$) is stable under the Gaussian approximation, but unstable under density evolution.

However, on the other hand, stability under density evolution fully guarantees stability under the Gaussian approximation. These two are the same if ρ_k=1 for some k≥2, and are very close if $\rho(x)$ is concentrated at two consecutive degrees. This is the criterion that we will use in the sequel to optimize the degree distribution for irregular LDPC codes under the Gaussian approximation in an AWGN channel.

8.5.2 The asymptotic bit error rate in large SNR under the Gaussian approximation

In this part we will derive the asymptotic bit error rate (BER) in large t_l (high SNR) by using BP algorithm under the Gaussian assumption in an AWGN channel.

(A) $0<\lambda_2<\lambda_2^*$: If $0<\lambda_2<\lambda_2^*$ (i_1=2) and for large t_l, based on (8-95) we can approximately formulate t_l as

$$t_l \approx c + 4l\log\left(\frac{\lambda_2^*}{\lambda_2}\right) \tag{8-101}$$

where c is a constant. Under the Gaussian approximation and all-zero information bits transmitted over an AWGN channel, the asymptotic bit error rate, corresponding to the variable node of degree i ($2 \le i \le d_v$), at the lth iteration is

$$\begin{aligned} P_l^{(i)} &= \lambda_i^{'}\, Q\left(\frac{s+it_l}{\sqrt{2\left(s+it_l\right)}}\right) \\ &= \lambda_i^{'}\, Q\left(\sqrt{\frac{s+it_l}{2}}\right) \end{aligned} \tag{8-102}$$

where

$$\lambda_i^{'} = \frac{\lambda_i / i}{\sum_{j=2}^{d_v} \lambda_j / j} \tag{8-103}$$

is the fraction of degree-i variable nodes. The average bit error rate at the lth iteration is

$$P_l = \sum_{i=2}^{d_v} \lambda_i^{'} Q\left(\sqrt{\frac{s+it_l}{2}}\right) \tag{8-104}$$

Since the first nonzero item in (8-104) becomes dominant when t_l is large in case of $0 < \lambda_2 < \lambda_2^*$, P_l can be further approximated as

$$P_l \approx \lambda_2^{'}\, Q\left(\sqrt{\frac{s+2t_l}{2}}\right) \tag{8-105}$$

Applying the identity

$$Q\left(\sqrt{\frac{x}{2}}\right) = \left(1+O\left(x^{-1}\right)\right) \frac{1}{\sqrt{\pi x}} \exp\left(-\frac{x}{4}\right) \tag{8-106}$$

and t_l in (8-101), we obtain

$$P_l \approx \frac{\lambda_2^{'}}{\sqrt{\pi\left(s+2t_l\right)}}\exp\left(-\frac{s+2t_l}{4}\right)$$

$$=\frac{\lambda_2^{'}}{\sqrt{\pi\left[\left(s+2c\right)+8l\log\left(\lambda_2^{*}/\lambda_2\right)\right]}}\exp\left(-\frac{\left(s+2c\right)+8l\log\left(\lambda_2^{*}/\lambda_2\right)}{4}\right)$$

$$=\frac{h_1}{\sqrt{h_2+l}}\left(\frac{\lambda_2}{\lambda_2^{*}}\right)^{2l} \tag{8-107}$$

where h_1 and h_2 are constants that depend on the degree distributions and s.

(B) λ_2=0 : For λ_2=0 (i_1>2) and large t_l, then based on (8-94) t_l is approximated as

$$t_l \approx d\left(i_1-1\right)^l \tag{8-108}$$

where d is a positive constant that depends on the degree distribution and s. Thus, again using the identity in (8-106), the asymptotic bit error rate dominated by the i_1th term at the lth iteration becomes

$$P_l \approx \lambda_{i_1}^{'}\,Q\left(\sqrt{\frac{s+i_1t_l}{2}}\right)\approx \lambda_{i_1}^{'}\,Q\left(\sqrt{\frac{s+i_1d\left(i_1-1\right)^l}{2}}\right)$$

$$\approx \frac{\lambda_{i_1}^{'}}{\sqrt{\pi\left(s+i_1d\left(i_1-1\right)^l\right)}}\exp\left(-\frac{s+i_1d\left(i_1-1\right)^l}{4}\right)$$

$$\approx \frac{\lambda_{i_1}^{'}\exp\left(-s/4\right)}{\sqrt{\pi i_1}\sqrt{\frac{s}{i_1d}+\left(i_1-1\right)^l}}\exp\left(-\frac{i_1d}{4}\left(i_1-1\right)^l\right)$$

$$\approx \frac{h_3}{\left(i_1-1\right)^{l/2}}\exp\left(-h_4\left(i_1-1\right)^l\right) \tag{8-109}$$

where h_3 and h_4 are constants that depend on the degree distributions and s.

8.6 Optimal Design of Irregular LDPC Codes Based on the Gaussian Approximation with BP Algorithm over AWGN Channel

In this chapter we will optimize the degree distributions for check and variable nodes under the Gaussian approximation with BP decoding algorithm in AWGN channel. The procedure is similar as the way for the optimization of irregular LDPC codes over BSC by linear programming, which is introduced in the previous chapter.

8.6.1 The optimization of degree distribution $\rho(x)$ for check nodes

Assuming that check node degree distribution $\lambda(x)$ and standard deviation σ_0 of band-limited Gaussian noise are given, we will optimize the variable node degree distribution $\rho(x)$ by maximizing the convergence rate of the bit error probability using linear programming. The constraints are 1) normalization: $\sum_{j=2}^{d_c}\rho_j = 1$; and 2) $\int_0^1 \rho(x)dx$ is a constant simply denoted by $\int\rho$. The objective function is to maximize the rate of convergence of error probability characterized by Δt in (8-94). Clearly, this objective function can be alternatively minimized by

$$\text{minimum} \sum_{j=2}^{d_c}\left[\rho_j \log(j-1)\right] \tag{8-110}$$

in case of ignoring the $O\left(t^{-1}\right)$ term when t is large enough.

Let us establish the standard linear program and its related dual program [64][65] in this case as follows

$$\text{minimize} \quad \mathbf{c}^T\mathbf{x} \tag{8-111a}$$

$$\text{subject to} \quad \mathbf{A}\mathbf{x}=\mathbf{b} \tag{8-111b}$$

$$\mathbf{x}\geq\mathbf{0} \tag{8-111c}$$

where the column vectors $\mathbf{x}^T$ to be determined and of nonnegative in each component is expressed as

$$\mathbf{x}^T=[\rho_2,\ \rho_3,\ \ldots,\ \rho_j,\ \rho_{j+1},\ \ldots,\ \rho_{d_c}] \tag{8-112}$$

and the matrix $\mathbf{A}$, the row vectors $\mathbf{c}^T$ and $\mathbf{b}^T$ are known and given as

$$\mathbf{A}=\begin{bmatrix} 1 & 1 & \cdots\cdot & 1 & 1 & \cdots\cdot & 1 \\ \frac{1}{2} & \frac{1}{3} & \cdots\cdot & \frac{1}{j} & \frac{1}{j+1} & \cdots\cdot & \frac{1}{d_c} \end{bmatrix} \tag{8-113a}$$

$$\mathbf{c}^T=[\log 1,\ \log 2,\ \ldots,\ \log(j-1),\ \log j,\ \ldots,\ \log(d_c-1)]$$

$$=[0,\ \log 2,\ \ldots,\ \log(j-1),\ \log j,\ \ldots,\ \log(d_c-1)] \tag{8-113b}$$

$$\mathbf{b}^T=[1,\ \textstyle\int\rho] \tag{8-113c}$$

Apparently, the rank of $\mathbf{A}$ is 2. The dual program to the primal linear program is

$$\text{maximize} \quad \boldsymbol{\beta}^T\mathbf{b} \tag{8-114a}$$

$$\text{subject to} \quad \boldsymbol{\beta}^T\mathbf{A}\leq\mathbf{c}^T \tag{8-114b}$$

where the row vector $\boldsymbol{\lambda}^T$ to be specified is

$$\boldsymbol{\beta}^T=[\beta_1,\ \beta_2] \tag{8-115}$$

(i) Find a basic feasible solution to the constraints of the primal program

Let $\mathbf{B}$ be the submatrix made up of two linearly independent columns of $\mathbf{A}$, i.e.,

$$\mathbf{B}=\begin{bmatrix} 1 & 1 \\ \frac{1}{j} & \frac{1}{j+1} \end{bmatrix} \tag{8-116}$$

where the degree index j is reasonablely determined in relation to the so-called average degree of check nodes as $j\le\left(\int\rho\right)^{-1}<j+1$, i.e.,

$$\frac{1}{j+1}<\int\rho\le\frac{1}{j} \tag{8-117}$$

Based on the Theorem 8 in the Appendix J, the two-dimensional row vector $\mathbf{x}_B^T=[\rho_j,\ \rho_{j+1}]$ corresponding to the basis $\mathbf{B}$ is

$$\begin{aligned}\mathbf{x}_B&=\mathbf{B}^{-1}\mathbf{b}\\ &=\begin{bmatrix} 1 & 1 \\ \frac{1}{j} & \frac{1}{j+1} \end{bmatrix}^{-1}\begin{bmatrix} 1 \\ \int\rho \end{bmatrix}\\ &=\frac{1}{\det(B)}\begin{bmatrix} \frac{1}{j+1} & -1 \\ -\frac{1}{j} & 1 \end{bmatrix}\begin{bmatrix} 1 \\ \int\rho \end{bmatrix}\end{aligned} \tag{8-118}$$

Hence, we obtain

$$\begin{aligned}\rho_j&=-\left(j(j+1)\right)\left(\frac{1}{j+1}-\int\rho\right)\\ &=j(j+1)\int\rho-j\end{aligned} \tag{8-119a}$$

$$\rho_{j+1}=-\big(j(j+1)\big)\left(-\frac{1}{j}+\int\rho\right)$$

$$=(j+1)-j(j+1)\int\rho=1-\rho_j \tag{8-119b}$$

It is easy to check that γ_i and γ_{i+1} are all nonnegative by (8-117) and we let $\rho_k=0$ for $k \notin \{j,\ j+1\}$. Hence, the basic feasible solution to the constraints of the primal program is

$$\rho(x)=\rho_j x^{j-1}+\rho_{j+1}x^j=\rho_j x^{j-1}+\left(1-\rho_j\right)x^j$$

$$=\left(j(j+1)\int\rho-j\right)x^{j-1}+\left((j+1)-j(j+1)\int\rho\right)x^j \tag{8-120}$$

(ii) Find a possible feasible solution $\boldsymbol{\beta}$ to the constraints of the dual program

Based on (8-114) the dual program of the primal program (8-111) is

$$\max\left\{\boldsymbol{\beta}^T\mathbf{b}=\beta_1+\beta_2\int\rho\,\middle|\,\beta_1+\frac{\beta_2}{k}\le\log(k-1),2\le k\le d_c\right\} \tag{8-121}$$

Now we can specify a possible feasible solution to the constraints of the dual program, which is in relation to the basic feasible solution in (8-119) for the primal program. The two components of $\mathbf{c}^T$ associated with the basis $\mathbf{B}$ forms the row vector $\mathbf{c}_B^T$=[$\log(j-1)$, $\log j$]. According to Theorem 8 in the Appendix J, the row $\boldsymbol{\beta}^T$ for the dual program is given as

$$\boldsymbol{\beta}^T=\mathbf{c}_B^T\mathbf{B}^{-1}$$

$$=\left[\log(j-1),\log j\right]\begin{bmatrix}\frac{1}{j+1} & -1\\ -\frac{1}{j} & 1\end{bmatrix}\frac{1}{\det(B)} \tag{8-122}$$

Thus, it results in

$$\beta_1 = (j+1)\log j - j\log(j-1) \tag{8-123a}$$

$$\beta_2 = j(j+1)\log\frac{j-1}{j} \tag{8-123b}$$

Clearly, $\boldsymbol{\beta}^T = \mathbf{c}_B^T\mathbf{B}^{-1}$ and $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$ satisfy the requirement that both objective functions have the same value, i.e., $\boldsymbol{\beta}^T\mathbf{b} = \mathbf{c}_B^T\mathbf{x}_B$.

(iii) Prove the obtained β to be the feasible solution to the constraints of the dual program

Finally, we will show that $\boldsymbol{\beta}$ fulfills the following constraints of the dual linear program in (8-114b) as

$$\beta_1 + \frac{\beta_2}{k} \le \log(k-1) \quad \text{for } k = 2, 3, \ldots, d_c \tag{8-124}$$

such that $\boldsymbol{\beta}$ becomes a feasible solution to the constraints of the dual program. Obviously, the inequality (8-124) holds exactly with equality for $k=j$ and $j+1$ based on the row vector $\boldsymbol{\beta}^T$ given by (8-122). Thus, we only need to prove that for $k \notin \{j,\ j+1\}$ the inequality also holds.

Let the function $\Delta(k,\beta_2)$ be the following equation

$$\begin{aligned}\Delta(k,\beta_2) &= \left(\log k - \frac{\beta_2}{k+1}\right) - \left(\log(k-1) - \frac{\beta_2}{k}\right) \\ &= \left(\log k - \frac{j(j+1)}{k+1}\log\frac{j-1}{j}\right) - \left(\log(k-1) - \frac{j(j+1)}{k}\log\frac{j-1}{j}\right)\end{aligned}$$

$$=\frac{1}{k(k+1)}\left[j(j+1)\log\frac{j-1}{j}-k(k+1)\log\frac{k-1}{k}\right] \tag{8-125}$$

It is easy to check that $\Delta(k,\beta_2)$ falls into following categories

$$\Delta(k,\beta_2):\begin{cases}<0 & if \quad 2\le k\le j-1\\ =0 & if \quad k=j\\ >0 & if \quad k\ge j+1\end{cases} \tag{8-126}$$

Thus, we get $\Delta(k,\beta_2)$ for $k=j$

$$\Delta(j,\beta_2)=\left(\log j-\frac{\beta_2}{j+1}\right)-\left(\log(j-1)-\frac{\beta_2}{j}\right)$$

$$=\beta_1-\beta_1=0 \tag{8-127}$$

(A) $k\ge j+1$:

For $k=j+1$ $\Delta(k,\beta_2)$ is expressed as

$$\Delta(j+1,\beta_2)=\left(\log(j+1)-\frac{\beta_2}{j+2}\right)-\left(\log j-\frac{\beta_2}{j+1}\right)$$

$$=\left(\log(j+1)-\frac{\beta_2}{j+2}\right)-\beta_1>0 \tag{8-128a}$$

We obtain

$$\beta_1+\frac{\beta_2}{j+2}<\log(j+1) \tag{8-128b}$$

Therefore, the inequality (8-124) holds for $k=j+2$.

For $k=j+2$ $\Delta(k,\beta_2)$ becomes

$$\Delta(j+2,\beta_2)$$

$$=\left(\log(j+2)-\frac{\beta_2}{j+3}\right)-\left(\log(j+1)-\frac{\beta_2}{j+2}\right)>0 \tag{8-129a}$$

Using (8-128b) it yields

$$\left(\log(j+2)-\frac{\beta_2}{j+3}\right)>\left(\log(j+1)-\frac{\beta_2}{j+2}\right)>\beta_1 \tag{8-129b}$$

We have

$$\beta_1+\frac{\beta_2}{j+3}<\log(j+2) \tag{8-129c}$$

Thus, the inequality (8-124) holds for $k=j+3$. Similarly, by induction we can show that it is also valid for $k\geq j+4$. □

(B) $k\leq j-1$:

For $k=j-1$ $\Delta(k,\beta_2)$ can be expressed as

$$\Delta(j-1,\beta_2)=\left(\log(j-1)-\frac{\beta_2}{j}\right)-\left(\log(j-2)-\frac{\beta_2}{j-1}\right)$$

$$=\beta_1-\left(\log(j-2)-\frac{\beta_2}{j-1}\right)<0 \tag{8-130a}$$

Equivalently, that is

$$\beta_1+\frac{\beta_2}{j-1}<\log(j-2) \tag{8-130b}$$

Therefore, the inequality (8-124) holds for $k=j-1$.

For $k=j-2$ $\Delta(k,\beta_2)$ becomes

$$\Delta(j-2,\beta_2)$$

$$=\left(\log(j-2)-\frac{\beta_2}{j-1}\right)-\left(\log(j-3)-\frac{\beta_2}{j-2}\right)<0 \qquad \text{(8-131a)}$$

Using (8-130b) we have

$$\beta_1<\left(\log(j-2)-\frac{\beta_2}{j-1}\right)<\left(\log(j-3)-\frac{\beta_2}{j-2}\right) \qquad \text{(8-131b)}$$

Equivalently, that is

$$\beta_1+\frac{\beta_2}{j-2}<\log(j-3) \qquad \text{(8-131c)}$$

Hence, the inequality (8-124) holds for $k=j-2$. Similarly, by induction we can show that it also holds for $2\le k\le j-3$. Finally, the constraints of the dual program in (8-1124) are all satisfied. □

Example 8.5: Consider an irregular LDPC codes ensemble, whose degree distribution for check nodes to be optimized is

$$\rho(x)=\frac{1}{4}x^2+\frac{1}{3}x^3+\frac{3}{8}x^4+\frac{1}{24}x^5 \qquad \text{(8-132)}$$

We have $\int\lambda=\int_0^1\rho(x)dx=\frac{179}{720}$. The average degree of variable nodes satisfy the inequality as $j\le\left(\int\rho\right)^{-1}<j+1$ with j=4. According to (8-119a) and (8-119b), the coefficients of the concentrated degree distribution for check nodes is

$$\rho_4^{(c)}=j(j+1)\int\rho-j$$

$$=4(4+1)\frac{179}{720}-4=\frac{35}{36} \qquad \text{(8-133a)}$$

$$\rho_5^{(c)} = (j+1) - j(j+1)\textstyle\int \rho$$

$$= 1 - \rho_4^{(c)} = \frac{1}{36} \tag{8-133b}$$

The *right-concentrated* degree distribution $\rho_c(x)$ under the Gaussian assumption in an AWGN channel is

$$\rho_c(x) = \rho_4^{(c)} x^3 + \rho_5^{(c)} x^4$$

$$= \frac{35}{36} x^3 + \frac{1}{36} x^4 \tag{8-134}$$

It is easy to check that $\int \rho_c = \int_0^1 \rho_c(x)dx = \frac{179}{720}$, which keeps the same value as $\int \rho$ associated with the variable nodes degree distribution without concentration processing. □

(iv) Simulaton results

So far we have shown that the *right-concentrated* degree with two nonzero consective items can maximize the rate of convergence of the error probability under the Gaussian assumption. However, we may still want to visualize how the optimization of $\rho(x)$ is performed by numerical simulations. Fig. 8.4 depicts a series of functions

$$f_j(s,t) - t = \phi^{-1}\left(1 - \left[1 - \sum_{i=2}^{d_v} \lambda_i \phi\big(s + (i-1)t\big)\right]^{j-1}\right) - t \tag{8-135a}$$

where j=2, 3,…., 10 and $\lambda(x)$ in (8-93a) is used for variable nodes and the squared noise power of Gaussian noise is equal to the Gaussian approximation threshold of the code, i.e., σ_0=0.9669 [16]. Fig. 8.5 shows the function independent of j

$$f(s,t)-t=\sum_{j=2}^{d_c}\rho_j f_j(s,t)-t \tag{8-135b}$$

where $\rho(x)$ in (8-93b) is used for check nodes. Figs. 8.4 and 8.5 demonstrate how two curves corresponding to j=8 and 9 in $\{f_j(s,t)-t\}$ are combined to produce the single curve $f(s,t)-t$ independent of j. Note that $f(s,t)-t$ is always above 0 and barely touches 0 at one point. If it hits 0, then the critical point t^* where the curve hits 0 becomes a fixed point, and the error probability cannot be decreased further.

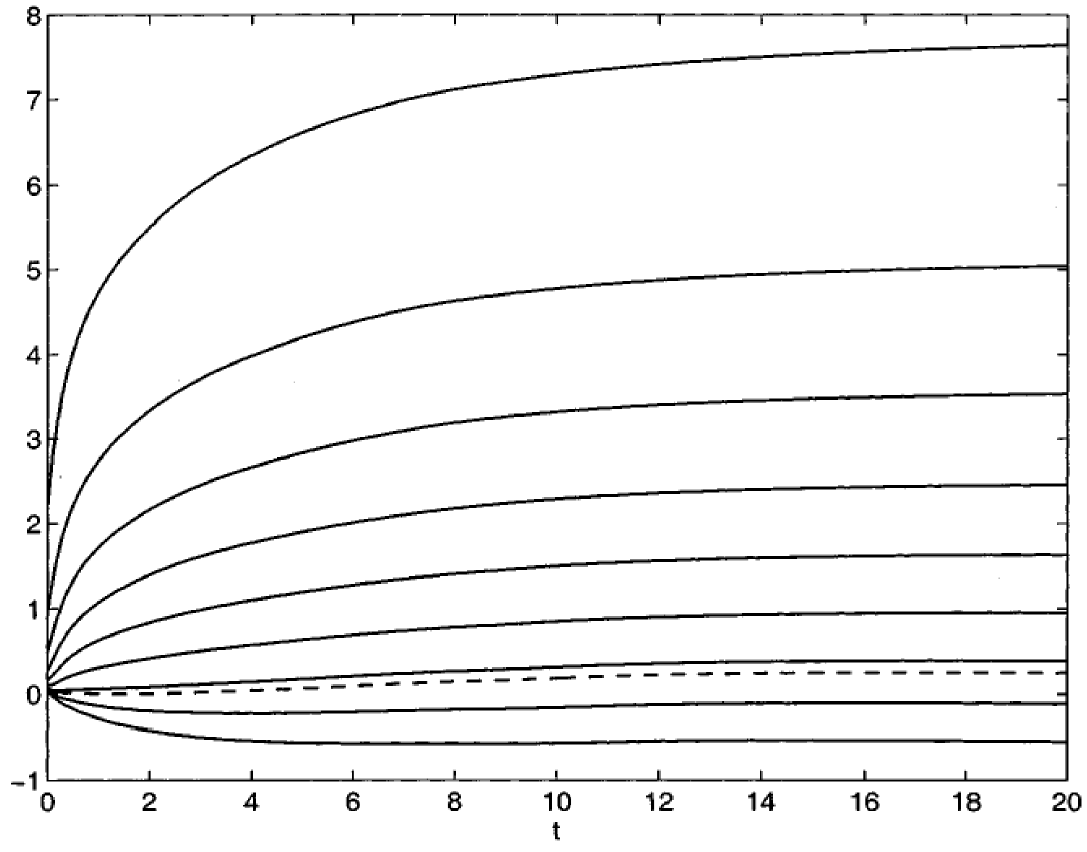


Fig.8.4 The functions $f_j(s,t)-t$ for j=2, 3, …., 10 depicted from top to bottoem by the solid lines and $f(s,t)-t$ depicted by the dashed lines.

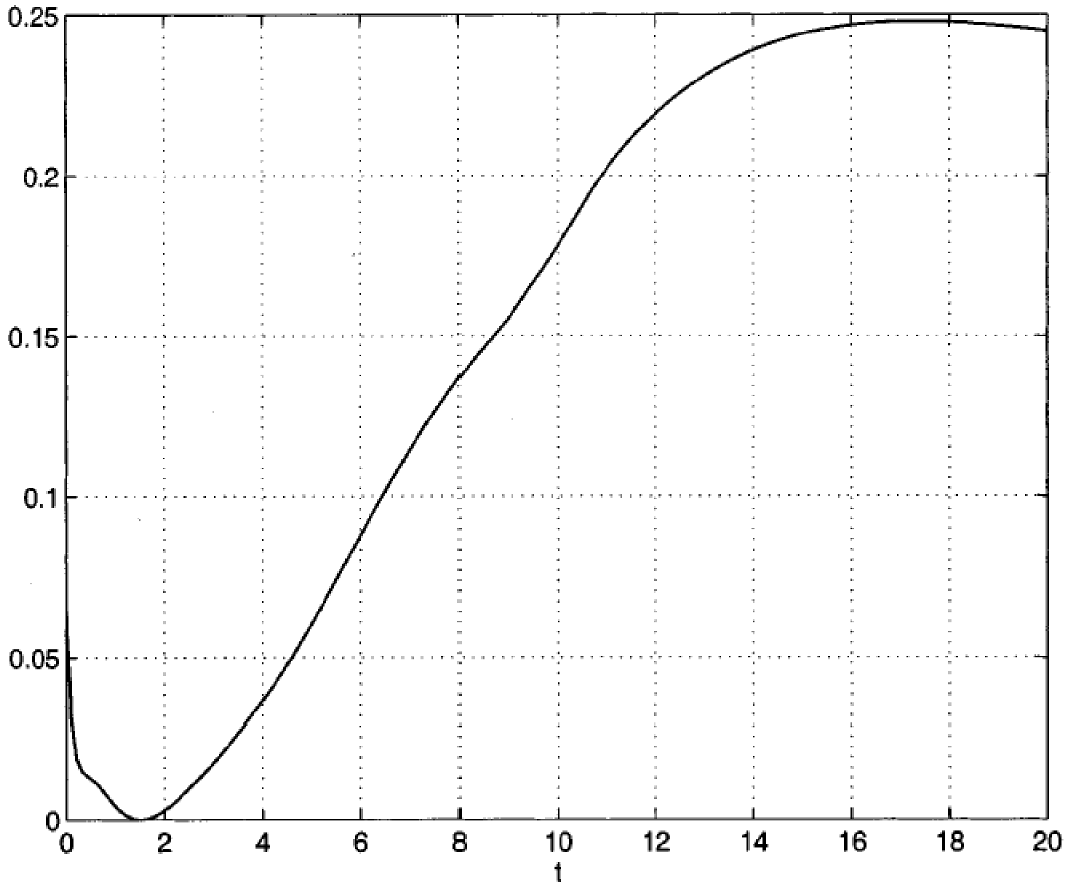


Fig. 8.5 $f(s,t)-t$ independent of j and as a function of t magnified.

8.6.2 The optimization of degree distribution $\lambda(x)$ for variable nodes

Similar as the optimiation of degree distribution $\rho(x)$ for check nodes, the optimization of $\lambda(x)$ for variable nodes can be also done by the assumptions that $\rho(x)$ and σ_0 are given. Based on the methods offered by [16], we optimize $\lambda(x)$ by maximizing the rate of the code as

$$r=1-\frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx} \tag{8-136}$$

The constraints are 1) normalization: $\sum_{i=2}^{d_v} \lambda_i$ =1; and 2) the inequality constraint in (8-92). This can be realized uing linear programming. Note that an inequality constraint can be converted to the corresponding equality constraint by introducting the so-called *slack variables*. The interested reader can refer it to [64][65][66].

Fig. 8.6 shows a series of functions

$$h_i(s,r)-r$$

$$=\phi\left(s+(i-1)\sum_{j=2}^{d_c}\rho_j\phi^{-1}\left(1-(1-r)^{j-1}\right)\right)-r \tag{8-137a}$$

where j=2, 3, …., 20 and $\rho(x)$ in (8-93b) is used for check nodes and the squared noise power of Gaussian noise is also equal to the Gaussian approximation threshold of the code, i.e., σ_0=0.9669 [16]. Fig. 8.7 demonstrates the function

$$h(s,r)-r$$

$$=\sum_{i=2}^{d_v}\lambda_i h_i(s,r)-r \tag{8-137b}$$

where $\lambda(x)$ in (8-93a) is used for check nodes. Figs. 8.6 and 8.7 show how each curve in { $h_i(s,r)-r$ } for i=2, 3, …., 20 is combined to generate the single curve $h(s,r)-r$ independent of i. Note that $h(s,r)-r$ is always below 0 and barely touches 0 at one point. If it hits 0, then the critical point t^* where the curve hits 0 becomes a fixed point, and the error probability cannot relaize further decreasing.

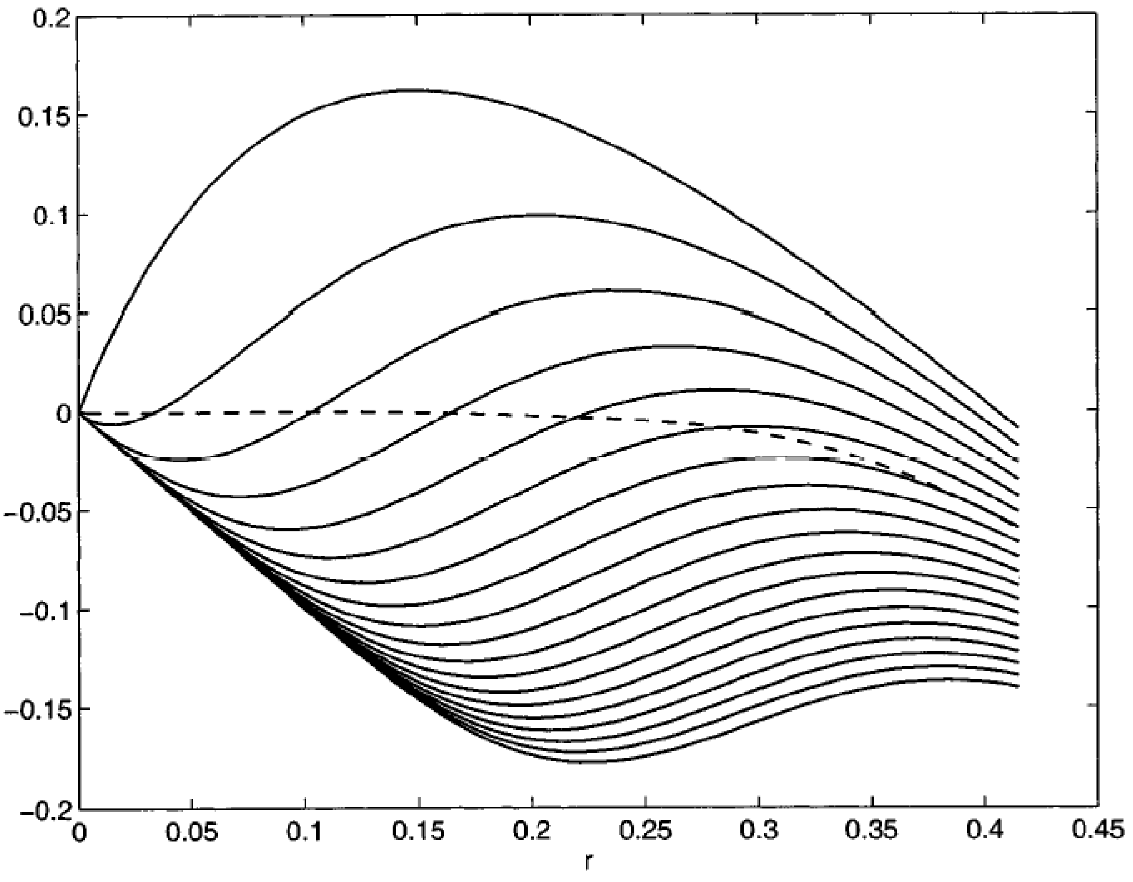


Fig. 8.6 The functions $h_i(s,r)-r$ for i=2, 3, …., 20 depicted from top to bottoem by the solid lines and $h(s,r)-r$ depicted by the dashed lines.

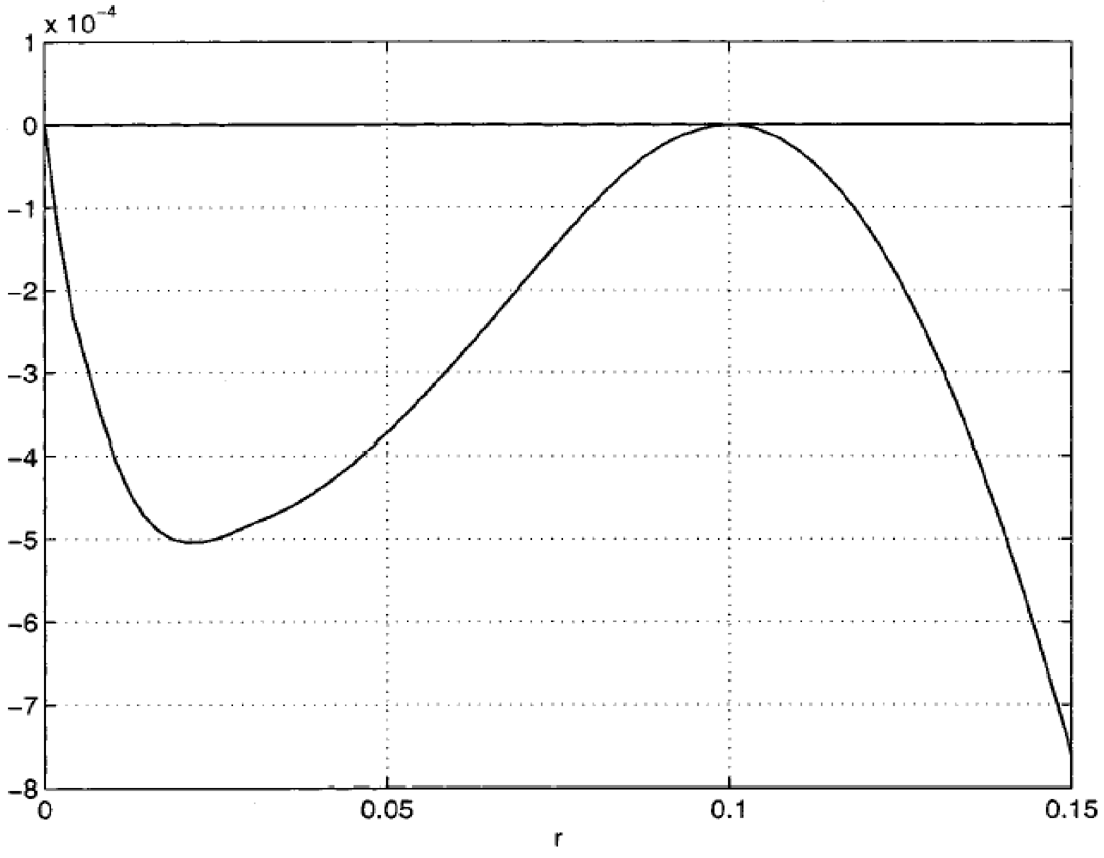


Fig. 8.7 $h(s,r)-r$ independent of i and as a function of r magnified.