A More Accurate One-Dimensional Analysis and Design of Irregular LDPC Codes

Masoud Ardakani, *Student Member, IEEE,* and Frank R. Kschischang, *Senior Member, IEEE*

***Abstract*—We introduce a new one-dimensional (1-D) analysis of low-density parity-check (LDPC) codes on additive white Gaussian noise channels which is significantly more accurate than similar 1-D methods. Our method assumes a Gaussian distribution in message-passing decoding only for messages from variable nodes to check nodes. Compared to existing work, which makes a Gaussian assumption both for messages from check nodes and from variable nodes, our method offers a significantly more accurate estimate of convergence behavior and threshold of convergence. Similar to previous work, the problem of designing irregular LDPC codes reduces to a linear programming problem. However, our method allows irregular code design in a wider range of rates without any limit on the maximum variable-node degree. We use our method to design irregular LDPC codes with rates greater than $1/4$ that perform within a few hundredths of a decibel from the Shannon limit. The designed codes perform almost as well as codes designed by density evolution.**

***Index Terms*—Degree distribution, density evolution, extrinsic information transfer (EXIT) charts, Gaussian assumption, irregular low-density parity-check (LDPC) codes.**

I. Introduction

THE design of irregular codes defined on graphs and determination of the convergence threshold of a given ensemble of codes for different decoding algorithms have been of great interest, e.g., [1]–[8]. The rediscovery of low-density parity-check (LDPC) codes (first introduced by Gallager [9]) by MacKay *et al.* [10], [11], drew attention to the analysis and design of these codes. We now know that LDPC codes are capable of approaching the capacity of many channels [5], [12] under message-passing decoding and, in fact, they can achieve the capacity of the binary erasure channel [13], [14].

The problem of analysis of LDPC codes is addressed by Richardson *et al.* in [4] and a density-evolution algorithm is proposed to find the convergence behavior of LDPC codes under various decoding schemes. Using density evolution, given the initial probability density function (pdf) of the log-likelihood ratio (LLR) messages, one can compute the pdf of LLR messages at any iteration, assuming a code of very long block length. As a result, one can test whether, for a given channel condition, the decoder converges to zero error probability or not. This allows for the design of irregular LDPC codes which perform very close to the Shannon limit using density evolution as a probe [5], [12], i.e., finding the convergence threshold of different irregular codes by density evolution and choosing the best one.

Density evolution may be unattractive for a few reasons. Finding a good degree sequence using density evolution requires intensive computations and/or a long search, since the optimization problem is not convex [5]; furthermore, it does not provide any insight in the design process, and it is intractable for some of the codes defined on graphs.

Another approach for finding convergence behavior of iterative decoders is to use extrinsic information transfer (EXIT) charts, e.g., [7], [8], [15], and [16]. Although this method is not as accurate as density evolution, its lower computational complexity and its reasonably good accuracy make it attractive. EXIT charts provide a one-dimensional (1-D) analysis, which allows one to visualize the convergence behavior of the decoder, and can reduce the irregular code optimization to a linear program [1] which is both faster and more insightful than density-evolution-based approaches. Moreover, this approach is applicable to many of the codes defined on graphs associated with iterative decoders.

There have been a number of approaches to 1-D analysis of LDPC codes [1], [15], [17]–[19], all of them based on the observation that the pdf of the decoder's LLR messages is approximately Gaussian. This approximation is quite accurate for messages sent from variable nodes, but less so for messages sent from check nodes. In this paper, we introduce a significantly more accurate 1-D analysis for LDPC codes. The key point is that, unlike previous approaches, we do not make the assumption that the check-to-variable messages have a Gaussian distribution. In other words, we assume a Gaussian distribution only for the channel messages and the messages from variable nodes, which in a real situation have a density very close to a Gaussian distribution. Under this assumption, we work with the "true" pdf of the messages sent from check nodes.

Previous work using a Gaussian approximation (GA) has serious limitations on the maximum node degree and also on the rate of the code. For example, in [1], the authors indicate that their analysis and design works for codes with a rate between 0.5 and 0.9, whose maximum variable-node degree is not greater than 10. Such a limitation on the maximum node degree prevents design of capacity-approaching degree distributions. In this paper, however, there is no limit on node degrees. In addition, it works for a wider range of code rates, with an effective design for codes with a rate greater than $1/4$.

The contributions of this paper are as follows: 1) We introduce a more accurate EXIT chart analysis of LDPC decoders

Paper approved by S. G. Wilson, the Editor for Multicarrier Modulation of the IEEE Communications Society. Manuscript received August 19, 2003; revised May 29, 2004. This paper was presented in part at the IEEE International Symposium on Information Theory, Lausanne, Switzerland, 2002.

The authors are with the Edward S. Rogers, Sr. Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 3G4, Canada (e-mail: masoud@comm.utoronto.ca; frank@comm.utoronto.ca).

Digital Object Identifier 10.1109/TCOMM.2004.838718

by relaxing some of the Gaussian assumptions on the LLR messages. 2) We compare different measures for making a GA, and show that using the mutual information as the metric for making the GA provides the most accurate results (similar result with the work of ten Brink for turbo codes [8]). 3) Within the framework of our approximation, we formulate the problem of designing irregular LDPC codes as a linear program.

This paper is presented as follows. In Section II, we propose our method for including a Gaussian assumption on messages. In Section III, we introduce EXIT charts. In Section IV, we use EXIT charts in the analysis of LDPC codes. The behavior of an iterative decoder for a regular LDPC code is investigated in this section. In Section V, we generalize the discussion to the irregular case. In Section VI, we address the problem of designing good irregular codes, and present a group of codes to verify the capabilities of our method. We conclude this paper in Section VII.

II. Gaussian Assumption on Messages

In [5], it has been shown that starting with a symmetric pdf, i.e., a pdf $f(x)$ that satisfies $f(x) = e^x f(-x)$, and using sum–product decoding, the pdf of LLR messages in the decoder remains symmetric.

A Gaussian pdf with mean m and variance σ^2 is symmetric if and only if $\sigma^2 = 2m$. As a result, a symmetric Gaussian density can be expressed by a single parameter. Interestingly, the density of LLR intrinsic messages for a Gaussian channel is symmetric, and hence, under sum–product decoding, remains symmetric.

Wiberg [20] noticed that the pdf of extrinsic LLR messages in an iterative decoder can be approximated by a Gaussian distribution. As a result, under a Gaussian assumption, a 1-D analysis of iterative decoders, i.e., tracking the evolution of a single parameter, becomes possible. In [1], based on a Gaussian assumption on all messages in the iterative decoder and the symmetry property, a 1-D analysis of LDPC codes has been carried out. In [18] and [19], again based on the symmetry property and a Gaussian assumption on all messages, together with an approximation on the input–output characteristics of check nodes, a 1-D analysis and design of LDPC codes and repeat-accumulate codes has been conducted.

In this section, we introduce our new way of including the Gaussian assumption on the messages that we refer to as the "semi-Gaussian" approximation.

A. Semi-Gaussian versus All-Gaussian Approximation

We consider an additive white Gaussian noise (AWGN) channel, whose input x is a random variable from alphabet $\{-1, +1\}$. We assume that the transmitter uses an LDPC code and the receiver decodes using the sum–product algorithm. We also assume that the information bit "0" is mapped to $+1$ on the channel, and that the all-zero codeword is transmitted (equivalent to the all-$(+1)$ channel word). Notice that since the channel is symmetric and sum–product decoding satisfies the check-node symmetry and variable-node symmetry conditions of [5], the decoder's performance is independent of the transmitted codeword.

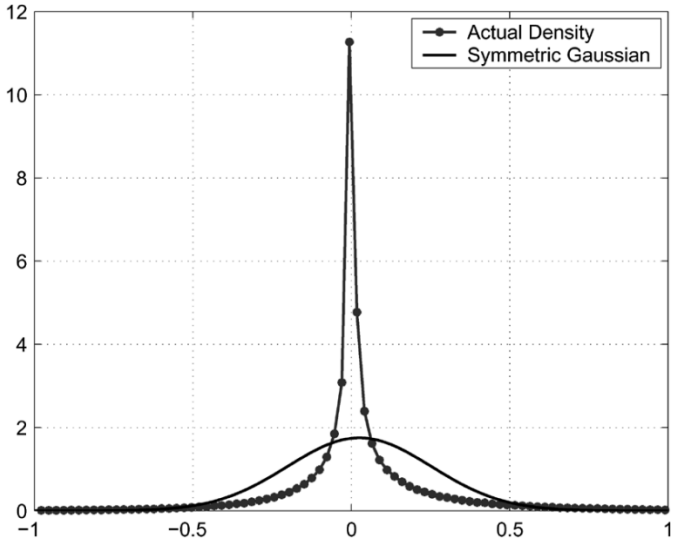

Fig. 1. Output density of a degree-six check node, fed with symmetric Gaussian density of -1 dB, is compared with the symmetric Gaussian density of the same mean.

In the LDPC decoder, there are three types of messages: the channel messages, the message from variable nodes to check nodes, and the messages from check nodes to variable nodes. We first study the channel messages. Each received symbol at the output of the channel, given transmission of the all-$(+1)$ channel word, is $z = x + n = 1 + n$, so its conditional pdf is

$$p(z|x=1) = \frac{1}{\sqrt{2\pi}\sigma_n} e^{-(z-1)^2/2\sigma_n^2}$$

where σ_n^2 is the variance of the Gaussian noise. The LLR value $\mathcal{L}$ corresponding to z is

$$\begin{aligned}\mathcal{L} &= \ln \frac{p(z|x=+1)}{p(z|x=-1)} = \frac{2}{\sigma_n^2} z \\ &= \frac{2}{\sigma_n^2}(1+n) \end{aligned} \tag{1}$$

which is a Gaussian random variable with

$$\mathrm{E}[\mathcal{L}] = m_0 = \frac{2}{\sigma_n^2}, \quad \mathrm{VAR}[\mathcal{L}] = \sigma_0^2 = \frac{4}{\sigma_n^2}. \tag{2}$$

Since the variance is twice the mean, the channel LLR messages have a symmetric Gaussian density.

For a symmetric Gaussian density with mean m and variance $\sigma^2 = 2m$, we define the signal-to-noise ratio (SNR) as m^2/σ^2. For a Gaussian channel with noise variance σ_n^2, we define SNR as $1/\sigma_n^2$. It can be easily verified that the SNR of the channel LLR messages, i.e., m_0^2/σ_0^2 is the same as the channel SNR.

Now we consider the variable-to-check and check-to-variable messages. At the first iteration the decoder receives only channel messages, but as soon as the messages pass through the check nodes, they lose their Gaussianity. Fig. 1 shows the output density of a check node of degree six when the input has a symmetric Gaussian density at SNR $= (m^2/\sigma^2) = -1$ dB. This shows that the pdf of the check-to-variable LLR messages can be very different from a Gaussian density. Notice that the magnitude of the output LLR message at a check node is less than the magnitude of all the incoming messages to that node, i.e., a check node acts as a soft "min" in magnitude, making the

density skewed toward the origin, hence, non-Gaussian. At the output of the variable nodes, however, it can be observed that the density is well approximated by a symmetric Gaussian, even when the inputs are highly non-Gaussian. This can be explained by the central limit theorem, since the update rule at a variable node is summation of incoming messages. As a result, for higher degree variable nodes, we expect even less error while making a Gaussian assumption.

A Gaussian assumption at the output of check nodes has two main problems. First, it is accurate only for check nodes of low degree at high SNR. This hinders analysis and design of low-rate codes, which are required to work at low SNR, and codes with a high check-node degree (very-high-rate codes). Second, it puts a limit on the maximum variable-node degree. To see why, assume that the mean of messages at the output of check nodes is m, and the mean of messages from the channel is m_0. Then the mean at the output of a degree d_v variable node will be $m_v = (d_v - 1)m + m_0$. As a result, the message-error probability at the output of this variable node, assuming a symmetric Gaussian density, will be

$$P_e = \frac{1}{2}\mathrm{erfc}(\sqrt{m_v}/2). \tag{3}$$

From (3), it follows that an error of Δm in approximation of m causes an error of ΔP_e in the approximation of error probability at the output of a degree-d_v variable node, which can be computed as

$$\Delta P_e = \frac{d_v - 1}{4\sqrt{\pi((d_v - 1)m + m_0)}} e^{-\frac{(d_v-1)m+m_0}{4}} \Delta m. \tag{4}$$

According to (4), in later iterations, when m is much larger than m_0, the higher the degree of the variable node, the lower the error in prediction of P_e, because the exponential term dominates the other term. However, in early iterations, when m_0 is dominant, the error in prediction of P_e increases almost linearly with d_v. In particular, at the first iteration when $m = 0$, ΔP_e is linear with $d_v - 1$. This puts a limit on the maximum variable degree allowed. In fact, as noted in [1], the previous method can be employed for designing irregular LDPC codes only with variable degrees less than or equal to 10.

The above facts have motivated us to remove the Gaussian assumption at the output of check nodes. In other work, the analysis proceeds in two steps: check-node analysis, which provides the input–output behavior of the decoder at the check nodes, and variable-node analysis, which provides the input–output behavior of the decoder at the variable nodes. At each step, the densities are forced to be Gaussian. To avoid a Gaussian assumption on the output of check nodes, we consider one whole iteration at once. That is to say, we study the input–output behavior of the decoder from the input of the iteration (messages from variable nodes to check nodes) to the output of that iteration (messages from variable nodes to check nodes). Fig. 2 illustrates the idea. In every iteration, we assume that the input and the output messages shown in Fig. 2, which are outputs of variable nodes, are symmetric Gaussian. We call our method the "semi-Gaussian," in contrast with the "all-Gaussian" methods that assume all the messages are Gaussian.

To analyze one iteration, we consider the depth-one tree of decoding, as shown in Fig. 2. We start with Gaussian-distributed

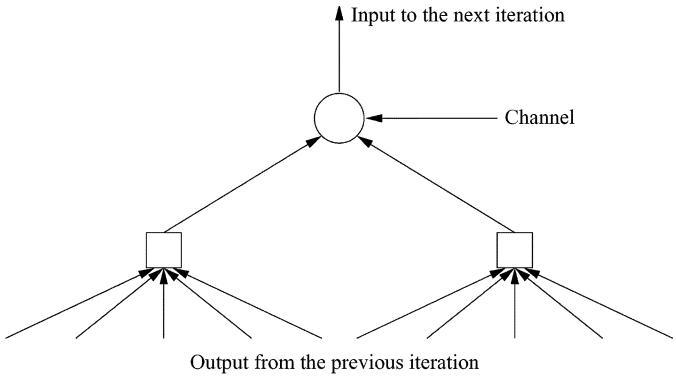

Fig. 2. Depth-one tree for a (3, 6) regular LDPC code.

messages at the input of the iteration and compute the pdf of messages at the output. This can be done by Monte Carlo simulation or "one-step" density evolution. Then we approximate the output pdf with a symmetric Gaussian.

The complexity of this analysis is significantly less than other methods based on density evolution. In density-evolution-based methods, we need the output density of one iteration in order to analyze the next iteration, making analysis of later iterations impossible without first analyzing the early iterations. However, a Gaussian assumption allows us to start from any stage of decoding. Using our method, one can do the analysis for only a few points of the EXIT chart and interpolate a curve on those points. Depending on the required accuracy, one may change the number of analyzed points. Close to capacity of the channel, this saves us a lot in complexity, since density evolution requires a large number of iterations to determine convergence.

III. EXIT Charts and Iterative Decoders

When the check and variable message updates are analyzed together, one iteration of LDPC decoding can be viewed as a black box that, in each iteration, uses two sources of knowledge about the transmitted codeword: the information from the channel (the intrinsic information) and the information from the previous iteration (the extrinsic information). From these two sources of information, the decoding algorithm attempts to get a better knowledge about the transmitted codeword, using this knowledge as the extrinsic information for the next iteration.

Now suppose U is some measure of knowledge about the transmitted codeword symbols, e.g., SNR, probability of error, mutual information,[1] and so on. In this setting, an EXIT chart based on U is a curve with two axes: U_{in} and U_{out}, and a parameter U_0, which is the information from the channel. Any point of this curve shows that if the knowledge from the previous iteration is U_{in}, using this together with the channel information U_0, the information at the output of this iteration would be U_{out}. Hence, this curve can be viewed as a function $U_{\mathrm{out}} = f(U_{\mathrm{in}}, U_0)$, where f depends on the code and the decoding scheme as well. For a fixed code, fixed decoding scheme, and fixed U_0, U_{out} is a function of just U_{in} or simply $U_{\mathrm{out}} = f(U_{\mathrm{in}})$.

[1]By mutual information, we mean the mutual information between a random variable representing a transmitted bit and another one representing a message that carries a belief about the transmitted bit.

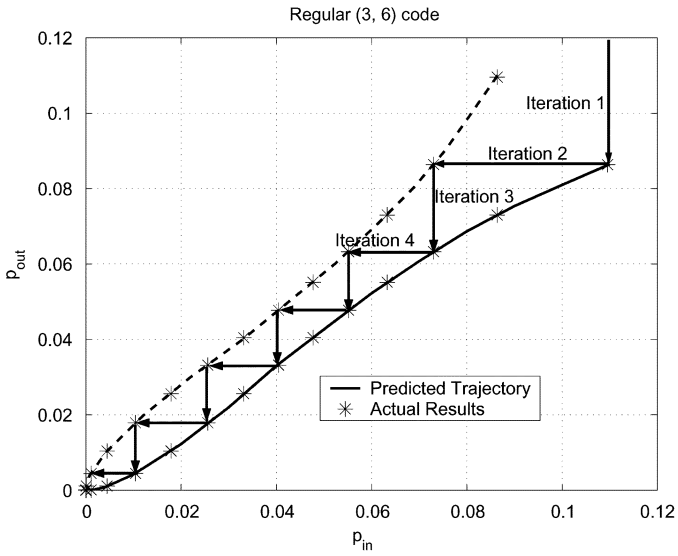

Fig. 3. EXIT chart, based on single-iteration analysis, comparing the predicted and actual decoding trajectories for a regular (3, 6) code of length 200 000.

Such an EXIT chart can be presented by plotting both f and its inverse f^{-1}. Presenting both f and f^{-1} on the same graph has the advantage of making the visualization of the decoder's behavior easier, as the output $U_{\rm out}$ from one iteration transfers to the input $U_{\rm in}$ of the next. Fig. 3, which is an EXIT chart based on message-error probability, shows the concept. Each arrow in Fig. 3 represents one iteration of decoding.

Many of the characteristics of decoding, such as the number of required iterations for a given target U and the threshold of convergence, can be predicted from EXIT charts. Note that if the "decoding tunnel" of an EXIT chart is closed, convergence does not occur. This makes it possible to predict the convergence threshold. The convergence threshold is the worst channel condition, measured by U_0, for which the tunnel is open.

It is worth mentioning that in the literature (e.g., [7], [8], [18], [19], [21], [22]) the term "EXIT chart" usually refers to taking U as mutual information, represented in $I_{\rm in}$ versus $I_{\rm out}$ axes. In this paper, however, by an EXIT chart, we mean *any* 1-D representation of the decoder's behavior. Of course, the metric chosen for making the GA affects the accuracy of the approximation. However, once an approximation is made, any parameter of the approximating density can be used to represent that density. Thus, for example, an EXIT chart can be computed by matching a symmetric Gaussian density to the true density based on a mutual information measure, but that EXIT chart may be plotted using, say, the probability of error of that Gaussian density.

Notice that one measure of a symmetric Gaussian density can be translated to another measure using explicit relations that they have with each other. For example, (3) allows translation of the mean of a symmetric Gaussian density to its error rate. Translation to mutual information requires a mapping function $J(\sigma)$, which maps standard deviation σ of a symmetric Gaussian density to its mutual information. It has been shown in [7] that

$$\mathcal{J}(\sigma) = 1 - \int_{-\infty}^{+\infty} \frac{e^{-(t-\sigma^2/2)^2/2\sigma^2}}{\sqrt{2\pi}\sigma} \cdot \log_2(1+e^{-t})\, dt. \quad (5)$$

Notice that $\mathcal{J}$ is a function of σ (and only σ), so it can be computed once and tabulated.

For turbo codes, in [15] and [16], some SNR-based measures are used to include the Gaussian assumption, while in [23], a combination of SNR and mutual information measures is used, and in [7] a pure mutual information measure is used. A comparison among different measures can be found in [8], which shows that for turbo codes, a GA based on matching mutual information predicts the behavior of iterative decoders more accurately than SNR-based GAs.

The choice of EXIT chart measure U also has a direct effect on the process of designing irregular LDPC codes, as will be discussed in Section VI.

IV. Analysis of Regular LDPC Codes

Our goal in this section is to find the EXIT chart of an iterative sum–product decoder for regular LDPC codes. Fig. 2 shows a depth-one tree factor graph for a regular LDPC code. Considering $U_{\rm in}$ for the messages from the previous iteration and U_0 for the channel message, under a Gaussian assumption, each of $U_{\rm in}$ and U_0 can be translated to corresponding Gaussian densities. Now, using density evolution, one can compute the output pdf, whose measure $U_{\rm out}$ can be computed. As a result, for a given U_0 and $U_{\rm in}$, $U_{\rm out}$ after one iteration is computed.

Define $U(f(x))$ as the function which takes a pdf $f(x)$ and returns its U measure, and define $\mathcal{G}(u)$ as the function which maps $u \in [U_{\min}, U_{\max}]$ to a symmetric Gaussian density $\mathcal{G}$, such that $U(\mathcal{G}) = u$. Now for a regular LDPC code with variable-node degree d_v and check-node degree d_c, we define $\mathcal{D}_{d_c,d_v}(f(x), U_0)$ as the mapping that takes an input pdf $f(x)$ and the channel condition U_0 and returns the output pdf after one iteration of density evolution on the depth-one tree of decoding. We can formulate our analysis as

$$U_{\rm out} = U\left(\mathcal{D}_{d_c,d_v}(\mathcal{G}(U_{\rm in}), U_0)\right). \quad (6)$$

Using (6), for a fixed U_0, a point of the $U_{\rm in} - U_{\rm out}$ curve is achieved. We can repeat this for as many points as we wish and interpolate between points to form our EXIT chart.

Fig. 3 shows our result for a regular (3, 6) code. It compares the actual results with the predicted trajectory using our method. One can see that there is very close agreement between our prediction and the actual results. The measure for making the GA is error probability. The EXIT chart is also represented based on error probability.

Table I shows the error in approximating the threshold of some regular codes using some different 1-D methods. The all-Gaussian column shows the error of the method of [1], while in all other columns, the Gaussian assumption is included only for the output of the variable nodes. The difference between these columns is that they use different measures for including the GA. A comparison among different measures shows that using mutual information as our measure of approximation gives rise to the most accurate results, in agreement with the results obtained in [8] for turbo codes.

For computation of the approximated threshold using an EXIT chart analysis, we need to vary the SNR and find the minimum value of SNR for which the EXIT chart is still open.

TABLE I
ERROR IN APPROXIMATION OF THRESHOLD USING DIFFERENT 1-D METHODS

d_v	d_c	Code Rate	Error in dB using				
			All-Gauss.	P of Error	Mean	Variance	Mutual Info.
3	4	0.25	0.103	0.036	0.154	0.523	0.014
3	5	0.4	0.079	0.018	0.117	0.447	0.007
3	6	0.5	0.061	0.012	0.099	0.379	0.005
4	8	0.5	0.055	0.006	0.078	0.281	0.005
5	10	0.5	0.028	0.005	0.059	0.212	0.004

TABLE II
ERROR IN APPROXIMATION OF MESSAGE-ERROR RATE AFTER THE FIRST ITERATION

d_v	d_c	Channel SNR (dB)	ΔP_e All-Gaussian %	ΔP_e Semi-Gaussian %
10	6	-3	10.17	0.02
10	6	-1	6.51	0.09
10	6	0	3.71	0.18
10	8	-1	9.08	0.02
10	10	-1	9.79	0.04
15	8	-1	13.58	0.01
20	8	-1	17.75	0.00
30	8	-1	25.25	0.00
30	8	$+1$	13.25	0.19

This can effectively be done through a binary search. To find the exact threshold, we perform a similar binary search, however, for each SNR, we use discretized density evolution [12] to verify convergence. As shown in [12], an 11-b discretization of messages will result in an approximation of threshold accurate to the third decimal digit. A 12-b discretization would be accurate to the fourth decimal point. In Table I, we have used 11-b discretization. Later in Table IV, where we need more accuracy, we use 12-b discretization.

According to Table I, a 1-D analysis can predict the threshold of convergence within 0.01 dB of the true value. For variable degrees greater than three, the error is in the order of a few thousandths of a decibel. The data in this table suggests that the all-Gaussian method approximates the threshold of convergence with an acceptable accuracy. However, an accurate prediction of the threshold of convergence cannot indicate the accuracy of a method completely. As we discussed earlier, the all-Gaussian method shows significant error when the variable degree is large and when the SNR is low (early iterations).

Table II compares the error in the approximation of message-error probability after first iteration in a depth-one tree of decoding using all-Gaussian and semi-Gaussian methods. We have chosen the first iteration because the all-Gaussian method shows more errors at the first iteration, when the mean of messages is minimum. For the semi-Gaussian method, we have used mutual information to approximate the density. The values of d_c and d_v in this table are typical values used for irregular codes. For regular codes, d_c is always greater than d_v, and we are not interested in large values for d_v. Table II shows that a Gaussian assumption at the output of check nodes can result in large errors when channel SNR is low and/or variable degree is high. The semi-Gaussian method, on the other hand, shows negligible error in all cases.

V. ANALYSIS OF IRREGULAR LDPC CODES

An irregular LDPC code is usually described by the fraction of edges incident to variable nodes and check nodes of different degrees. Following the notation used in [3], we specify an ensemble of irregular LDPC codes by its variable and check edge-degree distributions $\{\lambda_2, \lambda_3, \ldots\}$ and $\{\rho_2, \rho_3, \ldots\}$, where λ_i is the fraction of edges connected to variable nodes of degree i, and ρ_j is the fraction of edges connected to check nodes of degree j. In the remainder of this paper, by a variable/check-degree distribution, we mean a variable/check edge-degree distribution.

A single depth-one tree cannot be defined for irregular codes. If the check-degree distribution is fixed, each variable degree gives rise to its own depth-one tree. Irregularity in the check nodes is taken into account in these depth-one trees. For any fixed check-degree distribution, we refer to the depth-one tree associated with a degree i variable node as "degree i depth-one tree."

For reasons similar to the case of regular codes, we assume that, at the output of any depth-one tree, the pdf of LLR messages is well approximated with a symmetric Gaussian. As a result, the pdf of LLR messages at the input of check nodes can be approximated as a mixture of symmetric Gaussian densities. The weights of this mixture are determined by the variable-degree distribution. This pdf can be expressed as

$$\mathcal{G}_\lambda = \sum_{i\geq 2} \lambda_i \mathcal{G}(m_i, 2m_i) \tag{7}$$

where m_i is the mean of messages at the output of a degree i variable node, and $\mathcal{G}(m, \sigma^2)$ represents a Gaussian distribution with mean m and variance σ^2. It is also clear that

$$m_i = m_0 + (i-1)m_{\text{chk}} \tag{8}$$

where m_{chk} is the mean of messages at the output of check nodes and m_0 is the mean of intrinsic messages.

According to (7) and (8), for a Gaussian mixture with a given degree distribution $\lambda = \{\lambda_1, \lambda_2, \ldots\}$ and fixed channel condition, every m_{chk} can be translated to a unique mixed symmetric Gaussian density. In other words, given λ, the set of possible mixed symmetric Gaussian densities at the input of the iteration is 1-D, i.e., can be expressed by one parameter. This makes it possible to define a one-to-one relation from the set of real numbers to the set of pdfs, which relates each m_{chk} to its equivalent mixed Gaussian density. Since other metrics are strictly increasing or decreasing with m_{chk}, this one-to-one relation can be defined for them, as well. For instance, error probability is strictly decreasing with m_{chk}, hence, any value of error probability can be mapped to its equivalent m_{chk}, and so to its equivalent input pdf.

This one-to-one correspondence allows one to translate every U_{in} to a unique input pdf to the iteration. The input pdf together with one iteration of density evolution, on a degree i depth-one tree, gives the output pdf for a degree-i variable node. This pdf

can be well approximated by a symmetric Gaussian density and can be presented by $U_{\text{out},i}$.

A formulation similar to the case of regular codes can be made here. Define $\mathcal{G}_\lambda(u)$ as the function which maps $u \in [U_{\min}, U_{\max}]$ to a mixed symmetric Gaussian density $\mathcal{G}_\lambda$, with variable-degree distribution λ, such that $U(\mathcal{G}_\lambda) = u$. Now for a degree-$i$ depth-one tree of decoding with check-node degree distribution ρ, we define $\mathcal{D}_{\rho,i}(f(x), U_0)$ as the mapping that takes an input pdf $f(x)$ and the channel condition U_0 and returns the output pdf after one iteration of density evolution on the depth-one tree of decoding. We can formulate our analysis as

$$U_{\text{out},i} = U(\mathcal{D}_{\rho,i}(\mathcal{G}_\lambda(U_{\text{in}}), U_0)). \tag{9}$$

Repeating the same thing for all variable degrees present in the code, we can find $U_{\text{out},i}$ at the output of all variable-node degrees. Now, our task is to compute U_{out} for the mixture of all variable nodes. This task depends on the choice of U. For some choices of U, such as error probability, this is always simplified to a linear combination of $U_{\text{out},i}$ weighted by the variable-degree distribution. This is because using Bayes' rule, p_{out} can be computed as

$$\begin{aligned} p_{\text{out}} &= \sum_{i\geq 2} \Pr(\text{degree} = i) \cdot \Pr(\text{error}|\text{degree} = i) \\ &= \sum_{i\geq 2} \lambda_i \cdot p_{\text{out},i} \end{aligned} \tag{10}$$

where $p_{\text{out},i}$ is the already-computed message error rate at the output of the degree-i variable node. By computing p_{out}, one point of the EXIT chart of the irregular code is computed.

A similar formulation can be used for the mean of the messages [1]. It has been shown in [22] that when the messages have a symmetric pdf, mutual information combines linearly to form the overall mutual information.

In this paper, for analysis of the irregular case, we are most interested to represent the results in a $p_{\text{in}} - p_{\text{out}}$ axis. This is because it simplifies the analysis of the irregular case to the above-mentioned linear combination, and is more insightful than other parameters. Notice that in the phase of approximation, any other measure can be used. In the remainder of this paper, by an EXIT chart, we mean a p_{in} versus p_{out} EXIT chart, unless otherwise stated.

According to (10), the EXIT chart for an irregular code is a linear combination of EXIT charts for different variable degrees. As a result, we have a set of 1-D elementary curves whose linear combination explains the convergence behavior of the irregular code. Fig. 4 shows an example of these elementary curves.

A comparison between the predicted convergence behavior and the actual result, once using the semi-Gaussian and once using the all-Gaussian method, has been depicted in Fig. 5. As expected, the all-Gaussian method shows significantly larger error in early iterations. The semi-Gaussian method, however, is in close agreement with the actual results.

VI. Design of Irregular LDPC Codes

Design of irregular LDPC codes which can outperform regular codes was first considered in [3]. Different methods for designing irregular LDPC codes have been introduced and used

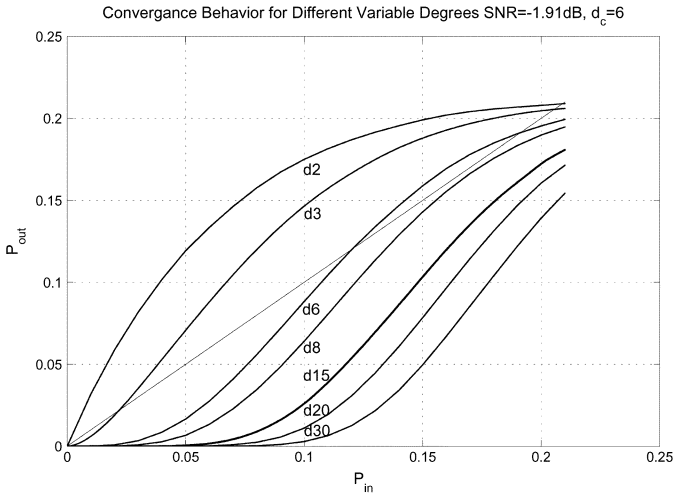

Fig. 4. EXIT charts for different variable degrees on an AWGN channel at -1.91 dB when $d_c = 6$.

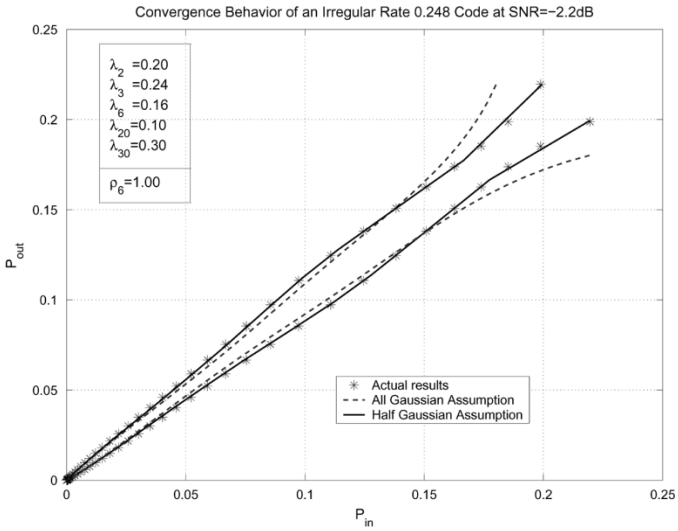

Fig. 5. Actual convergence behavior of an irregular code compared with the predicted trajectory using the semi-Gaussian and the all-Gaussian methods.

[1], [5], [12]. In this paper, EXIT charts are designed (shaped) to guarantee good performance of the code. EXIT charts are even used in the design of good turbo codes. For example, [21] presents a rate-$1/2$ turbo code which approaches the Shannon limit to within 0.1 dB.

In our 1-D framework, the design problem is simplified to a linear program. Fixing the check-degree distribution, we refer to the EXIT chart of a code whose variable nodes are all of degree i by $f_i(x)$, and we call it an "elementary" EXIT chart. This name has been chosen as opposed to the EXIT chart for an irregular code, which is a mixture of elementary EXIT charts. According to (10), the EXIT chart of an irregular code with the variable-degree distribution $\lambda = \{\lambda_2, \lambda_3, \ldots\}$, can be written as

$$f(x) = \sum_{i\geq 2} \lambda_i f_i(x).$$

This fact simplifies the problem of designing an irregular code to the problem of shaping an EXIT chart out of a group of elementary EXIT charts. We wish to maximize the rate of the code

while having the EXIT chart of the irregular code satisfy the condition that $f(x) < x$ for all $x \in (0, p_0]$, where p_0 is the initial message-error rate at the decoder. Being below x for all x guarantees convergence to zero message-error rate for an infinite block-length code.

The design rate of the code is $R = 1 - (\sum \rho_j/j / \sum \lambda_i/i)$, and hence, for a fixed check-degree distribution, the design problem can be formulated as the following linear program:

$$\begin{aligned} \text{maximize} \quad & \sum_{i\geq 2} \lambda_i/i \\ \text{subject to} \quad & \lambda_i \geq 0, \quad \sum_{i\geq 2} \lambda_i = 1, \quad \text{and} \\ & \forall p_{\text{in}} \in (0, p_0] \left(\sum_{i\geq 2} \lambda_i f_i(p_{\text{in}}) < p_{\text{in}} \right). \end{aligned}$$

In the above formulation, we have assumed that the elementary EXIT charts are given. In practice, to find these curves, we need to know the degree distribution of the code. We need the degree distribution to associate every input p_{in} to its equivalent input pdf. In other words, prior to the design, the degree distribution is not known, and as a result, we cannot find the elementary EXIT charts to solve the linear program above.

To solve this problem, we suggest a recursive solution. At first we assume that the input message to the iteration, Fig. 2, has a single symmetric Gaussian density instead of a Gaussian mixture. Using this assumption, we can map every message-error rate at the input of the iteration to a unique input pdf, and so find $f_i(x)$ curves for different i. It is interesting that even with this assumption, the error in approximating the threshold of convergence, based on our observations, is less than 0.3 dB, and the codes which are designed have a convergence threshold of at most 0.4 dB worse than those designed by density evolution. One reason for this is that when the input of a check node is mixture of symmetric Gaussians, due to the computation at the check node, its output is dominated by the Gaussian in the mixture having the smallest mean.

After finding the appropriate degree distribution based on a single Gaussian assumption, we use this degree distribution for finding the correct elementary EXIT charts based on a Gaussian mixture assumption. Now we use the corrected curves to design an irregular code. In this level of design, the designed degree distribution is close to the used degree distribution in finding the elementary EXIT charts. Therefore, analyzing this code with its actual degree distribution shows a minor error. One can continue these recursions for higher accuracy. However, in our examples, after one iteration of design, the designed threshold and the exact threshold differed less than 0.01 dB. So we propose the following design algorithm.

Algorithm
1. Map every p_{in} to the associated symmetric Gaussian density (a single Gaussian).
2. Find the EXIT charts for different variable degrees.
3. Find a linear combination with an open EXIT chart that maximizes the rate and meets all the required design criterion.

TABLE III
IRREGULAR CODES DESIGNED BY SEMI-GAUSSIAN METHOD FOR GAUSSIAN CHANNEL WITH SNR $= -1.91$ dB

Degree Sequence	Based on Single Gaussian	Based on Gaussian Mixture
λ_2	0.2754	0.2669
λ_3	0.1647	0.2123
λ_6	0.1331	0.1094
λ_8	0.1026	0.1441
λ_9	0.0829	0.0263
λ_{15}	0.0175	0.0284
λ_{20}	0.0348	0.0277
λ_{30}	0.1890	0.1849
ρ_6	1.0000	1.0000
Code Rate	0.3227	0.3407
Gap to the Shannon Limit (dB)	0.486	0.139

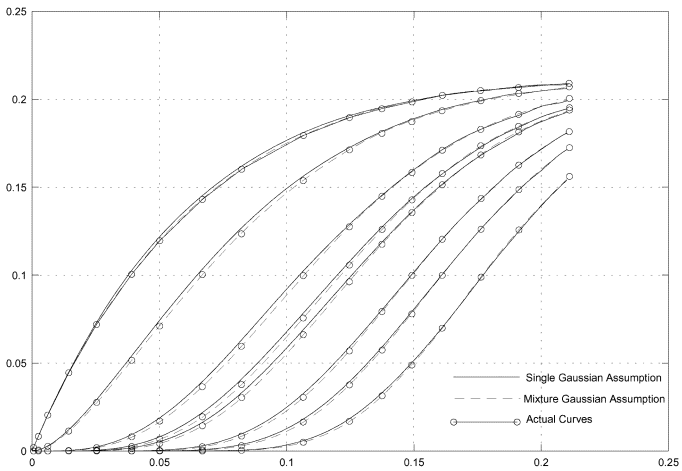

Fig. 6. Illustrating the improved accuracy of elementary exit charts based on a Gaussian mixture, compared with a single Gaussian.

4. Use this degree sequence for mapping p_{in} to the appropriate input densities (a Gaussian mixture).
5. Find the EXIT charts for different variable degrees.
6. Find a linear combination with an open EXIT chart that maximizes the rate and meets all the required design criterion.

A. Examples and Simulation Results

As an example, we consider designing an irregular code with the maximum possible rate for a channel at SNR$= (1/\sigma_n^2) = -1.91$ dB. We also want the code to have regular check degree and use no variable degree greater than 30. These restrictions constrain the decoding complexity to a practically acceptable level. As noticed in previous work [1], [5], the check side of the best codes designed using density evolution is either regular or concentrated around one degree. Usually, we choose the average check-node degree and we build this average using only two check degrees. Chung *et al.* provide guidelines for choosing the average check degree [1].

TABLE IV
LIST OF IRREGULAR CODES DESIGNED BY SEMI-GAUSSIAN METHOD

Degree Sequence	Code1	Code2	Code3	Code4	Code5	Code6
$d_{v_1}, \lambda_{d_{v_1}}$	2, .1786	2, .1530	2, .1439	2, .1890	2, .2444	2, .3000
$d_{v_2}, \lambda_{d_{v_2}}$	3, .3046	3, .2438	3, .1602	3, .1158	3, .1687	3, .1937
$d_{v_3}, \lambda_{d_{v_3}}$	5, .0414	7, .1063	5, .1277	4, .1153	4, .0130	4, .0192
$d_{v_4}, \lambda_{d_{v_4}}$	6, .0531	10, .2262	6, .0219	6, .0519	5, .1088	7, .2378
$d_{v_5}, \lambda_{d_{v_5}}$	7, .0007	14, .0305	7, .0279	7, .0875	7, .1120	14, .0158
$d_{v_6}, \lambda_{d_{v_6}}$	10, .4216	19, .0001	8, .0103	14, .0823	14, .1130	15, .0114
$d_{v_7}, \lambda_{d_{v_7}}$	-	23, .1293	12, .1551	15, .0007	15, .0577	20, .0910
$d_{v_8}, \lambda_{d_{v_8}}$	-	32, .0736	30, .0004	16, .0001	25, .0063	25, .0002
$d_{v_9}, \lambda_{d_{v_9}}$	-	38, .0372	37, .3525	39, .3573	25, .0109	30, .0232
$d_{v_{10}}, \lambda_{d_{v_{10}}}$	-	-	40, .0001	40, .0001	40, .1652	40, .1077
d_c	40	24	22	10	7	5
Convergence threshold (Channel's σ_n)	0.5072	0.6208	0.6719	0.9700	1.1422	1.5476
Rate	0.9001	0.7984	0.7506	0.4954	0.3949	0.2403
Gap to the Shannon limit (dB)	0.1308	0.0630	0.0666	0.1331	0.1255	0.2160

For this example, by studying the elementary EXIT charts for different values of check-node degree, it can be seen that the check-node degree must be six. Now, using the curves in Fig. 4, which are computed based on a single Gaussian assumption at the input of check nodes, we design a code using linear programming. The designed variable-degree sequence is presented in Table III. Having the degree distribution of this code, we find the exact elementary EXIT charts for each variable degree. Fig. 6 compares the exact curves with the approximated curves under a single Gaussian assumption. Using exact curves, we repeat the design and achieve a new degree sequence. Since the curves used in the design are based on another degree sequence, we have compared the actual curves for this final design with the curves which are, in fact, used for this design in Fig. 6. This figure shows that the approximation at this level of design is highly accurate.

According to Table III, the designed code has a gap of about 0.14 dB from the Shannon limit. The best codes designed using density evolution with maximum variable degree of 30 have a gap of about 0.09 dB from the Shannon limit when the rate of the code is 0.5 [5]. Unfortunately, to the best of our knowledge, there is no published rate-1/3 code designed by density evolution. So, a more fair comparison is not possible here. Nevertheless, this comparison shows that our 1-D approach does about as well as methods based on density evolution under any practical measure.

We have used our design method to design a number of irregular codes with a variety of rates. The results are presented in Table IV. In the design of the presented codes, we have had some additional constraints, which makes the designed codes more practical for implementation. We have avoided any variable or check node with degrees higher than 40. We have limited the number of used degrees to 10, favoring results which use fewer check/variable degrees. For check nodes, there has been no intention to make it regular, but as long as it has been possible to design a code with a regular check side with a performance loss less than 0.01 dB, we have favored a regular structure. Interestingly, all the codes have a regular structure at the check side. In the design of each code, we have maximized the code rate for a given channel SNR.

Table IV suggests that for code rates less than 0.25, the method shows noticeable inaccuracy. However, for a wide range of rates, it is quite successful. For rates greater than 0.85, getting close to capacity requires high-degree check nodes. To show that our method can actually handle high-rate codes, we designed a rate-0.9497 code, which uses check nodes of degree 120 but no variable node of degree greater than 40. The variable-degree sequence for this code is $\lambda = \{\lambda_2 = 0.1029, \lambda_3 = 0.1823, \lambda_6 = 0.1697, \lambda_7 = 0.0008, \lambda_9 = 0.1094, \lambda_{15} = 0.0240, \lambda_{35} = 0.2576, \lambda_{40} = 0.1533\}$. The threshold of this code is at $\sigma_n = 0.4462$, which means it has a gap of only 0.0340 dB from the Shannon limit. We have not tried our method for rates greater than 0.95, however, our other results suggest that the method should work fine for even higher rates.

VII. CONCLUSION

We have proposed a more accurate 1-D analysis for LDPC codes over the AWGN channel which is based on a Gaussian assumption at the output of variable nodes. We eliminate a Gaussian assumption at the output of check nodes, making our analysis significantly more accurate than previous approaches. Compared to the previous work on 1-D analysis of LDPC codes, our method not only offers better precision, but it can

also be used over a wider range of rates, channel SNRs, and maximum variable degrees.

We compared different measures for including the Gaussian assumption and showed that the mutual information measure allows us to analyze the convergence behavior of LDPC codes very accurately and predict the convergence threshold within a few thousandths of a decibel from the actual threshold.

We used EXIT charts based on message-error rate to design irregular LDPC codes. Our 1-D analysis gives a deep insight to each level of design, and the designed code performs as well as codes designed using methods based on density evolution. Since our analysis is not based on simplicity of check nodes, we believe it can be used for designing other codes defined on graphs, such as irregular turbo codes [6], repeat-accumulate codes [24], concatenated tree codes [25], or when more complicated check codes are employed in an LDPC structure.

REFERENCES

[1] S.-Y. Chung, T. J. Richardson, and R. L. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Inform. Theory*, vol. 47, pp. 657–670, Feb. 2001.

[2] M. G. Luby and M. Mitzenmacher, "Efficient erasure correcting codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 569–584, Feb. 2001.

[3] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. Inform. Theory*, vol. 47, pp. 585–598, Feb. 2001.

[4] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619–637, Feb. 2001.

[5] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599–618, Feb. 2001.

[6] B. J. Frey and D. J. C. MacKay, "Irregular turbo codes," in *Proc. 37th Allerton Conf. Communications, Control, Computing*, Allerton House, IL, 1999, pp. 241–248.

[7] S. ten Brink, "Iterative decoding trajectories of parallel concatenated codes," in *Proc. 3rd IEE/ITG Conf. Source, Channel Coding*, Munich, Germany, Jan. 2000, pp. 75–80.

[8] ——, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, pp. 1727–1737, Oct. 2001.

[9] R. G. Gallager, *Low Density Parity Check Codes*. Cambridge, MA: MIT Press, 1963.

[10] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *IEEE Electron. Lett.*, vol. 32, pp. 1645–1646, Aug. 1996.

[11] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, Mar. 1999.

[12] S.-Y. Chung, G. D. Forney, Jr., T. J. Richardson, and R. L. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, pp. 58–60, Feb. 2001.

[13] A. Shokrollahi, "New sequence of linear time erasure codes approaching the channel capacity," in *Proc. Int. Symp. Applied Algebra, Algebraic Algorithms, Error-Correcting Codes*. New York: Springer-Verlag, 1999, Lecture Notes in Computer Science, pp. 65–67.

[14] ——, "Capacity-achieving sequences," *IMA Vol. Math., Applicat.*, vol. 123, pp. 153–166, 2000.

[15] D. Divsalar, S. Dolinar, and F. Pollara, "Low complexity turbo-like codes," in *Proc. 2nd Int. Symp. Turbo Codes, Related Topics*, Brest, France, 2000, pp. 73–80.

[16] H. El Gamal and A. R. Hammons, "Analyzing the turbo decoder using the Gaussian approximation," *IEEE Trans. Inform. Theory*, vol. 47, pp. 671–686, Feb. 2001.

[17] F. Lehmann and G. M. Maggio, "An approximate analytical model of the message passing decoder of LDPC codes," in *Proc. IEEE Int. Symp. Information Theory*, Lausanne, Switzerland, July 2002, p. 31.

[18] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. Commun.*, vol. 52, pp. 670–678, Apr. 2004.

[19] S. ten Brink and G. Kramer, "Design of repeat-accumulate codes for iterative detection and decoding," *IEEE Trans. Signal Processing*, vol. 51, pp. 2764–2772, Nov. 2003.

[20] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Linköping Univ., Linköping, Sweden, 1996.

[21] S. ten Brink, "Rate one-half code for approaching the Shannon limit by 0.1 dB," *IEEE Electron. Lett.*, vol. 36, pp. 1293–1294, July 2000.

[22] M. Tuechler and J. Hagenauer, "EXIT charts and irregular codes," in *Proc. Conf. Information Sciences, Systems*, Princeton, NJ, Mar. 2002, pp. 748–753.

[23] M. Peleg, I. Sason, S. Shamai, and A. Elia, "On interleaved differentially encoded convolutional codes," *IEEE Trans. Inform. Theory*, vol. 45, pp. 2572–2582, Nov. 1999.

[24] H. Jin, A. Khandekar, and R. McEliece, "Irregular repeat-accumulate codes," in *Proc. 2nd Int. Symp. Turbo Codes, Related Topics*, Brest, France, 2000, pp. 1–8.

[25] L. Ping and K. Y. Wu, "Concatenated tree codes: A low-complexity high-performance approach," *IEEE Trans. Inform. Theory*, vol. 47, pp. 791–799, Feb. 2001.

Masoud Ardakani (S'01) received the B.Sc. degree in electrical engineering from Isfahan University of Technology, Isfahan, Iran, in 1994, and the M.Sc. degree in electrical engineering from University of Tehran, Tehran, Iran, in 1997. He is currently working toward the Ph.D. degree at the University of Toronto, Toronto, ON, Canada.

From 1997 to 1999, he was with the Electrical and Computer Engineering Research Center, Isfahan, Iran. His research interests are in the general area of digital communications, error-control coding, especially codes defined on graphs associated with iterative decoding, and MIMO systems.

Mr. Ardakani was the recipient of a number of scholarships and awards including the Edward S. Rogers Sr. Scholarship while at the University of Toronto.

Frank R. Kschischang (S'83–M'91–SM'00) received the B.A.Sc. degree (with honors) from the University of British Columbia, Vancouver, BC, Canada, in 1985 and the M.A.Sc. and Ph.D. degrees from the University of Toronto, Toronto, ON, Canada, in 1988 and 1991, respectively, all in electrical engineering.

He is a Professor of Electrical and Computer Engineering and Canada Research Chair in Communication Algorithms at the University of Toronto, where he has been a faculty member since 1991. During 1997–1998, he spent a sabbatical year as a Visiting Scientist at the Massachusetts Institute of Technology (MIT), Cambridge. His research interests are focused on the area of coding techniques, primarily on soft-decision decoding algorithms, trellis structure of codes, codes defined on graphs, and iterative decoders. He has taught graduate courses in coding theory, information theory, and data transmission.

Dr. Kschischang was the recipient of the Ontario Premier's Research Excellence Award. From October 1997 to October 2000, he served as an Associate Editor for Coding Theory for the IEEE TRANSACTIONS ON INFORMATION THEORY. He also served as technical program co-chair for the 2004 IEEE International Symposium on Information Theory held in Chicago, IL.