Finite-Length Scaling of Turbo-Like Code Ensembles on the Binary Erasure Channel

Iryna Andriyanova, *Member, IEEE*

***Abstract*— A possibility of estimating the finite-length performance of sparse-graph code ensembles gives two opportunities: to compare different codes of the same length in a context very close to real, practical applications and to perform the parameter optimization for a given code length [2]. We need a finite-length approximation that is valid for any code ensemble. The scaling approach seems to be a tool, general enough to provide such an approximation. However, the analytical derivation of parameters of the scaling approximation has been successful only for LDPC codes [1]; despite several attempts [25], [20], no such result was proposed for other code ensembles.**

In this paper, we focus on the finite-length performance of turbo-like codes, by applying the scaling approach to this case. In particular, by assuming the transmission over the binary erasure channel, we conjecture the scaling law and derive its scaling parameter. As examples, we present the performance estimation for Repeat-Accumulate codes [11], parallel turbo codes [8] and TLDPC codes [5], in all cases matching well the numerical results.

***Index Terms*— Iterative decoding, turbo-like codes, finite-length performance, binary erasure channel.**

I. Introduction

Sparse-graph codes are very attractive for a large range of communication systems applications, due to their close-to-capacity behaviour under iterative decoding of a reasonably low complexity. The asymptotic analysis of iterative decoding is well developed. It gives rise to a linear programming optimization of the parameters of a code ensemble, with the aim of obtaining the iterative decoding threshold as close as possible to the theoretical limit. However, the optimized parameters are not necessarily optimal in the finite-length setting: obtaining the best iterative decoding threshold does not imply obtaining the best slope of the ensemble error curve. Hence, the asymptotic optimization is not sufficient for design of sparse-graph codes of moderate codelengths, and some finite-length approximation taking into account the slope of the error curve is required.

The question which arises at this point is how to estimate the finite-length performance for sparse-graph codes. The error curve of a code ensemble can be seen as a sum of two contributions: the error-floor curve (the erasure-floor curve on the erasure channel) and the waterfall curve. The error-floor curve takes into account the errors of sublinear size. Typically such errors come from the local configurations in the code structure, which makes the decoder converge to an erroneous decision. Those configurations are called stopping sets on the erasure channel and trapping sets in general. Therefore, to estimate the error-floor curve we enumerate the trapping (stopping) sets and apply the union bound. In the literature, there are several interesting works presenting such an estimation for different families of sparse-graph codes, on the binary erasure channel [10], [16], [19], [21], [28], as well as on other channels [24].

The waterfall curve is given by errors whose sizes are linear in the codelength. They are caused by a non-zero correlation between the messages exchanged during the iterative decoding. The scaling approach was recently proposed for the waterfall curve estimation and was successfully applied for irregular LDPC codes over the BEC. In [1], [2], the authors present the form of the scaling law and a method to compute explicitly its parameters. Moreover, they present the linear-programming algorithm to optimize the parameters of the LDPC code ensemble as a function of the target error probability for a given value of the channel noise parameter. In [9], the proof of the scaling law for Poisson LDPC ensembles on the BEC is given.

Two research directions arise from the aforementioned results. The first is to generalize it to binary memoryless symmetric channels, other than the BEC. Step one of this generalization was first presented in [13] and further developed in [14]. It concerns the waterfall curve estimation for LDPC codes (regular and irregular).

According to the general scaling hypothesis [15], one supposes that, in the waterfall region, the error probability of any sparse-graph ensemble follows a scaling law. Hence, another research direction is the generalization of the scaling approach for different code ensembles. We are interested in extending the known methods for LDPC codes to other sparse-graph ensembles, in particular to those which can be decoded in a turbo-like way, such as Repeat-Accumulate (RA) codes [11], [17], turbo codes [8], Tail-biting Trellis LDPC codes [6], Zigzag-Hadamard codes [22].

In this work, some code structures, different from LDPC codes, are investigated. We consider turbo-like ensembles that allow for one-dimensional optimization of their parameters. Assuming that the transmission takes place on the BEC, we conjecture the scaling law for them, and we show the method of calculating its scaling parameters.

We also want to mention two previous attempts that have been made in the similar direction. The first work [25] concerns the computation of scaling parameters for Repeat-

Manuscript received October 1, 2008; revised January 15, 2009.

This work was presented in part at the IEEE International Zurich Seminar on Communications 2008 and the IEEE International Symposium on Information Theory 2008 in Toronto.

Iryna Andriyanova is with the ETIS laboratory at the ENSEA, Cergy-Pontoise, 95014 France (e-mail: iryna.andriyanova@ensea.fr). This research was accomplished when the author was with the LTHC laboratory at the EPFL, Switzerland.

Accumulate codes over the binary erasure channel, by basing on the covariance evolution. Unfortunately, the estimations, obtained in [25], do not match numerical results for short block lengths. The second work [20] gives an estimation of the block-error rate of parallel turbo codes, also over the binary erasure channel. The estimation is based on a scaling function, the parameter of which can be calculated through the computation of correlations between the messages at each iteration of the decoding. However, the analytical computation of the aforementioned parameter is not available and has to be done numerically.

The paper is organized as follows. We give the necessary details about the turbo-like codes in Section III and discuss the form of its scaling law in Section IV. We present the details of the derivation of the scaling parameter in Sections V and VI. Section VII is devoted to the numerical estimation of the shift parameter. In Section VIII we compare numerical results for different turbo-like ensembles with the estimations calculated using the scaling approach. Finally, we conclude the paper by discussion in Section IX.

This work is partly presented in the conference papers [4] and [3].

II. Notation

Symbols representing vectors are marked in bold, matrices are denoted by uppercase letters, their elements - by square brackets, transposition - by $(.)^T$. $\times$ or . denotes the matrix multiplication, $*$ - the element-wise matrix multiplication. We omit the superscript of the iteration number when we assume the asymptotic case, i.e. that it goes to infinity.

III. Turbo-Like Ensembles

In this section we give the necessary notions about turbo-like codes by presenting their structural particularity and the iterative decoding algorithm and the density evolution calculation.

A. Structure

Any sparse-graph code ensemble on some length N can be presented by means of some base code and of a bipartite graph. Let us define what the base code is.

Definition 1 (Base Code): The base code is a linear code of length M or a juxtaposition of several codes of total length M. We say that the base code has a convolutional structure if it can be seen as the serial concatenation of small identical codes $\mathcal{C}$. Such small codes are called *local* codes.

For instance, a convolutional base code is represented by the rectangle in Fig.1. Its Tanner graph is a chain of nodes corresponding to r local codes $\mathcal{C}$ in the structure of the base code. White circles at the end of the chain represent the initial and the final state nodes. It is also worth to notice that a convolutional base code has a time-invariant trellis representation, where a trellis section corresponds to a local code. We will use this representation later on in the paper. For more details about trellises see, for example, [18].

We are ready to make the following definition.

Definition 2 (Turbo-Like Sparse-Graph Code): Let $\mathcal{C}_B$be a convolutional base code of length M, having r local codes in its structure. Then the structure of a turbo-like code $\mathcal{C}_{TL}$ of length $N < M$ is represented by a Tanner graph with N variable nodes and a convolutional node corresponding to $\mathcal{C}_B$, such that there are $\lambda_i N$ edges connected to variable nodes of degree i, $i = 1 \dots i_{max}$, $\sum_{i=1}^{i_{max}} \lambda_i = 1$. Each variable node of degree i is randomly connected to i positions of the convolutional node. The obtained turbo-like code $\mathcal{C}_{TL}$ is the set of all the binary assignments of variable nodes such that the words induced from variable nodes to the convolutional node in the Tanner graph form codewords of $\mathcal{C}_B$.

The Tanner graph of a turbo-like code is given in Fig. 1. Black circles represent variable nodes and the rectangle - the convolutional node.

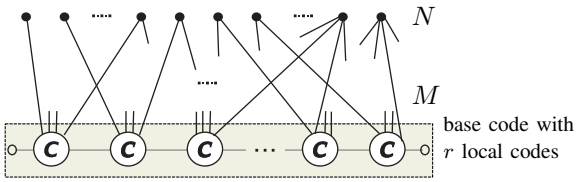

Fig. 1. Tanner graph of a turbo-like code.

Let us recall some notation to describe sparse-graph codes. **Notation** : We define the *degree distribution of variable nodes from the edge perspective* as

$$\lambda(x) = \sum_{i=1}^{i_{max}} \lambda_i x^{i-1}.$$

Also, we define the *degree distribution of edges of degrees higher than* 1 to be

$$\tilde{\lambda}(x) = \sum_{i>1} \tilde{\lambda}_i x^{i-1} = \sum_{i>1} \left(\frac{\lambda_i}{\sum_{i>1}, \lambda_i} \right) x^{i-1}$$

and their *average degree* as

$$\bar{\lambda} = \frac{1}{\sum_{i=2}^{i_{max}} \tilde{\lambda}_i / i}.$$

Definition 3 (Turbo-Like Ensemble): A turbo-like code ensemble of length N, parametrized by $\lambda(x)$ and $\mathcal{C}_B$, is the set of all the turbo-like codes with the degree distribution $\lambda(x)$ and the base code $\mathcal{C}_B$.

Below we give several examples of turbo-like codes.

Example [(c, d) Repeat-Accumulate codes (RA)] (c, d) RA codes can be seen as a class of turbo-like codes. Their base code is usually called the accumulator code. It contains parity codes of length $d + 2$ as local codes. The Tanner graph of the accumulator code is given in Fig.2. Unfilled circles in the figure represent state nodes which connect parity nodes.

The variable nodes in the RA Tanner graph have degrees c and 1, degree-1 variable nodes are connected to the state nodes and degree-c variable nodes are connected to parity nodes, $\lambda(x) = \lambda_1 + \lambda_c x^{c-1}$. Variable nodes of degree c correspond to information bits and those of degree 1 - to redundancy bits in the code structure. $\Diamond$

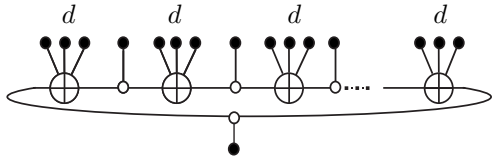

Fig. 2. Tanner graph of the accumulator code. $\oplus$ are parity nodes of degree d and $\circ$ are state nodes.

Example [Parallel turbo codes] The Tanner graph of parallel turbo codes contain two convolutional nodes and variable nodes of degree 2 and 1, therefore, $\lambda(x) = \lambda_1 + \lambda_2 x$. Variable nodes of degree 2 correspond to information bits and those of degree 1 - to redundancy bits in the code structure. $\Diamond$

Example [Tail-biting Trellis LDPC codes (TLDPC)] The structure of the TLDPC base code is represented in Figure 3. Its Tanner graph contains a cycle formed by check and state nodes, as well as other check nodes connected to the cycle. Hence, we distinguish two types of check nodes, those within the cycle and those outside of it. The degree of the i-th check node of the first or of the second type is denoted by a_i or b_i correspondingly, $i = 0, \ldots, r-1$; $\sum_i (a_i + b_i) = M$. It is simple to verify that the parity equation for a check node in the cycle is given by

$$s_{i+1} + s_i + \sum_{j=1}^{b_i} x_j = 0$$

and for a check node outside of it

$$s_i + \sum_{j=1}^{a_i} x_j = 0,$$

where x's and s's are the corresponding variables of the variable nodes and the state nodes. Note that TLDPC codes may have a non-zero fraction λ_1. In the particular case when all the a_i are equal to 1 and the degree of the variable nodes connected to check nodes of the first type is exactly 1, we obtain the Repeat-Accumulate code structure. $\Diamond$

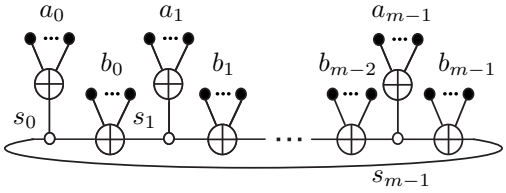

Fig. 3. Tanner graph of the TLDPC base code.

B. Turbo Scheduling

The iterative decoding for turbo-like codes is performed as follows. At the initial phase, the a priori probabilities of the code bits (variable nodes) are computed. Each decoding iteration then consists of two steps, corresponding respectively to the computation of *extrinsic* and *intrinsic* messages:

1) Computation of the extrinsic probabilities of bits of degrees > 1 at the convolutional node(s), usually with the help of the BCJR algorithm [7], and then they are passed to the corresponding variables nodes;
2) Computation of intrinsic probabilities at the variable nodes of degree > 1 and passing them to the convolutional node(s).

At the end of the decoding process, a posteriori probabilities at the variable nodes of degree > 1 are calculated.

C. Concentration and Density Evolution

The density evolution technique serves to study the iterative decoder behaviour in the average ensemble case, in the limit of large blocklengths. In the following, we consider a turbo-like ensemble and suppose that the transmission takes place over the binary erasure channel (BEC). We focus on the fixed-point equation of the density evolution for such an ensemble.

Assume that the all-zero codeword was transmitted and that the estimations obtained in the decoding process are independent. The latter is not true in general, because the extrinsic probabilities computed for two positions of the same convolutional code for some finite length are not independent. However, this technical difficulty is overcome by considering the window BCJR decoding algorithm, exactly in the same way as it has been proposed for turbo codes in Chapter 6.4 of [27]. Given such a decoding algorithm for convolutional codes, we can show the concentration result for turbo-like ensembles. As the proof of the concentration follows the lines of the proof done for LDPC codes [26], we will be brief here and will simply make the statement:

Theorem 1 (Concentration for turbo-like codes): Let a code C, chosen randomly from the turbo-like ensemble T of length N, be used for the transmission over the BEC with the erasure probability ϵ. Then the erasure probability $\mathbf{P}(C, \epsilon, l)$ after l iterations of iterative decoding satisfies

$$\mathbf{P}\{|\mathbf{P}(C, \epsilon, l) - \mathbb{E}_{C' \in T[N]}[\mathbf{P}(C', \epsilon, l)]| > \delta\} \leq e^{-\alpha N}$$

for any given $\delta > 0$, where α is a positive parameter not depending on N.

Now the ensemble average can be used as an indicator of the behaviour of individual codes in a given turbo-like family. Let us focus on the average erasure probabilities at the fixed point, i.e. when the number of iterations goes to infinity, at the limit of large blocklengths. To do this, we essentially follow the lines of the discussion in [23], generalizing it to turbo-like ensembles. Consider a turbo-like code of infinite length. Assume the transmission takes place over the BEC with the erasure probability ϵ. Perform the iterative decoding until the decoder reaches the fixed point, i.e. the extrinsic and intrinsic probabilities do not change with iterations. Denote by x (or y) the corresponding average erasure probability at the input (or at the output) of the base code decoder[1]. Notice that $y = y(x, \epsilon)$ is determined by the structure of the convolutional code. Then the following holds:

$$x - \epsilon \tilde{\lambda}(y(x, \epsilon)) = 0 \tag{1}$$

To be more complete, let us recall how to determine $y(x, \epsilon)$. We consider the BCJR decoding on the convolutional trellis, where each trellis section corresponds to a local code in the Tanner graph of Figure 1. Denote by $\mathbf{f}_i$ (or by $\mathbf{b}_i$) the forward (or backward) state of the trellis section i. Any state can be represented as a vector of length 2^m, $m \geq 1$, and, due

[1] x and y are usually called intrinsic and extrinsic probabilities

to the transmission over the BEC, is of special form: it has exactly 2^k non-zero entries, each of them being equal to 2^{-k}, $0 \leq k \leq m$. The number of all possible special forms for the state vectors is $\sum_{k=0}^{m} \binom{m}{k}_2$, where $\binom{\cdot}{\cdot}_2$ is the 2-Gaussian binomial coefficient. Let S be the set of all those special forms under the all-zero codeword assumption, and let its cardinality be p, so that $S = \{\mathbf{s}_1, \ldots, \mathbf{s}_p\}$. To each section i of the convolutional trellis, we associate the forward and backward transition-probability matrices F_i and B_i with elements

$$\begin{aligned} F_i[u,v] &= \mathbf{P}(\mathbf{f}_{i+1}^{(l)} = \mathbf{s}_u | \mathbf{f}_i^{(l)} = \mathbf{s}_v, \epsilon, x), \\ B_i[u,v] &= \mathbf{P}(\mathbf{b}_i^{(l)} = \mathbf{s}_u | \mathbf{b}_{i+1}^{(l)} = \mathbf{s}_v, \epsilon, x), \end{aligned}$$

$u, v = 1, \ldots, p$. Note that F_i and B_i are irreducible aperiodic stochastic matrices, and they have unique stationary vectors, which we denote by $\mathbf{f}$ and $\mathbf{b}$ correspondingly.

Let each trellis section carries s positions of degree > 1. Denote by e_j, $j = 1 \ldots s$, the event that the extrinsic message from the j-th position is erased. Define s matrices, $T_1 \ldots T_s$, with following elements:

$$T_j[u,v] = \mathbf{P}(e_j | \mathbf{f}[u], \mathbf{b}[v], \epsilon, x), \quad j = 1 \ldots s,$$

$u, v = 1, \ldots, p$. The average extrinsic probability is thus equal to

$$y(x, \epsilon) = \frac{1}{s} \sum_{j=1}^{s} \mathbf{f}^T . T_j . \mathbf{b}.$$

The iterative decoding threshold $\epsilon^\star$ is defined as the largest erasure probability below which the equation (1) has a unique solution $x = 0$. The average intrinsic (or extrinsic) probability at the decoding threshold is denoted by as $x^\star$ (or $y^\star$).

IV. Scaling Law

In the waterfall region the scaling approach seems to be a good tool for predicting the behaviour of the error probability and it gives very good results when applied to LDPC codes. The scaling analysis is based on the dynamics of the decoding process, i.e. on evolution of the decoder state with iterations. According to the general scaling hypothesis [15], one supposes that, in the waterfall region, the error-probability of any sparse-graph code ensemble follows a scaling law.

For finite-length turbo-like codes, as for any sparse-graph codes in general, one can follow the process of decoding by observing the number of not yet revealed variable nodes and the number of still erased intrinsic messages, exactly as it is described in Section III of [14]. We will not repeat the derivation of [14] here. It is nevertheless important to mention that we make two following assumptions:

Assumption 1: A given turbo-like ensemble is *unconditionally stable*, i.e. its iterative threshold is not determined by the stability condition. For instance, this is the case for turbo codes, RA codes and for the most of TLDPC codes presented in literature.

Assumption 2: Consider a transmission over the binary erasure channel with the help of a code, randomly chosen from some given turbo-like ensemble. Let $H^\star$ be the random variable[2] which denotes the number of not revealed variable nodes at the moment of the decoding process when the average intrinsic erasure probability is $x^\star$. Then the distribution of $H^\star$ is assumed to be Gaussian with mean $\epsilon^\star$.

By taking into account the assumptions and by following the derivation in [14], we obtain the two following conjectures for turbo-like ensembles:

Conjecture 1 (Basic Scaling Law for Turbo-like Codes): Assume the transmission was made over the BEC with the erasure probability ϵ. Then the average block erasure probability P_B for a given turbo-like ensemble of length N having n bits of degree > 1 is approximated by

$$P_B \simeq Q\left(\frac{\sqrt{n}(\epsilon^\star - \epsilon)}{\alpha} \right), \tag{2}$$

where $\epsilon^\star$ is the iterative decoding threshold and α is the scaling parameter depending only on the code ensemble.

Conjecture 2 (Refined Scaling Law for Turbo-like Codes): Assume the transmission was made over the BEC with the erasure probability ϵ. Then the average block erasure probability P_B for a given turbo-like ensemble of length N having n bits of degree > 1 is approximated by

$$P_B \simeq Q\left(\frac{\sqrt{n}(\epsilon^\star - \beta n^{-2/3} - \epsilon)}{\alpha} \right), \tag{3}$$

where $\epsilon^\star$ is the iterative decoding threshold, α and β are the scaling and the shift parameters which only depend on the code ensemble.

In summary, the conjectures are similar to those given in the case of LDPC codes [1], due to the fact that the scaling law is supposed to hold for all sparse-graph codes (over several assumptions). However, the expressions of parameters of the scaling law for turbo-like codes are not the same, because of the different structure of the base code. The main result of this paper consists of analytical derivation of the scaling parameter α for turbo-like code ensembles. We also give a simple yet efficient procedure of estimation of the shift parameter β, which is based on results given in [1] and [12].

V. Derivation of the Scaling Parameter α

The expression of α is computed as follows [14]:

$$\alpha = \frac{\partial^2 \epsilon(x)}{\partial x^2}|_\star \lim_{\epsilon \to \epsilon^\star} (x - x^\star) \sqrt{\frac{\nu}{\bar{\lambda}}}; \tag{4}$$

$\epsilon(x)$ is implicitly defined by the density evolution equation (1) and $(\epsilon^\star, x^\star)$ is the critical point of the equation which is assumed to be unique for simplicity[3]. ν in (4) is the variance of the number of erased messages coming out of variable nodes of degree > 1 for $\epsilon > \epsilon^\star$, calculated at the limit of large codelengths when the number of iterations goes to infinity. Thus, if $m^{(l)}$ is the number of erased messages coming out from variable nodes of degrees > 1 at the l-th decoding iteration, the variance is calculated as

$$\nu = \lim_{l \to \infty} \lim_{n \to \infty} \frac{\mathbb{E}[(m^{(l)})^2] - \mathbb{E}[m^{(l)}]^2}{\bar{\lambda} n}. \tag{5}$$

Notice that in order to compute α, we need to compute $\frac{\partial^2 \epsilon(x)}{\partial x^2}|_\star$, ν and then take the limit $\epsilon \to \epsilon^\star$. The most difficult

[2] $H^\star$ is equivalent to $\hat{H}_{x^*}$ in [14].

[3] The example of more that one critical point is presented in Section VIII-C

part here is to compute ν, and the main result of this paper is essentially the determination of this parameter for turbo-like codes. The shift parameter β still should be estimated numerically, although we need to do it only once for a given code ensemble.

It turns out that for turbo-like ensembles over the BEC α can be defined explicitly. Here is the statement of our main result:

Lemma 1 (Variance of number of erased messages): Assume a turbo-like ensemble of infinite length. For any $\epsilon > \epsilon^\star$, when $l \to \infty$,

$$\nu = \frac{(\epsilon\tilde{\lambda}'(y))^2\gamma}{(1-\epsilon\tilde{\lambda}'(y)p_C)^2} + O(1-\epsilon\tilde{\lambda}'(y)p_C)^{-1} \qquad (6)$$

where x and y are intrinsic and extrinsic erasure probabilities at the corresponding critical point,

$$\gamma = (1-y)p_{TE} - yp_{ELR} \qquad (7)$$

and the quantities p_C, p_{TE} and p_{ELR} are given by equations (15), (16) and (17) respectively.

Remark $\epsilon\tilde{\lambda}'(y)p_C \to 1$ while $\epsilon \to \epsilon^\star$, and in order to compute α according to (6), we only need to take into account the first term as it is dominant.

VI. Proof of Lemma 1

Now we focus on the computation of the variance of the number of erased messages ν in the computation tree for a code from given turbo-like ensemble. Previously, in the case of LDPC codes, the components of the computation tree were variable nodes and check nodes. Notice that they now are variable nodes of degree > 1 and convolutional nodes. The main difficulty in this case lies in determining the joint erasure probability for two arbitrary positions in a convolutional code. In the calculation of ν, we will proceed in two steps:

- first, we will study the erasure events for two positions of a convolutional code during the BCJR decoding.
- then we will compute the variance ν by generalizing the method that has been applied to LDPC codes in [2].

A. Erasure Propagation in Convolutional Trellises

For our purpose, we consider two arbitrarily chosen positions in a convolutional trellis with degrees > 1[4]. To determine the relation between intrinsic or extrinsic messages at two positions, we define the following type t:

Definition 4 (Relation between positions in a trellis): For two positions of degree > 1 in a convolutional trellis, corresponding to local codes C_i and C_j of the convolutional code, we put into correspondence the triplet $\mathbf{t} = (k, d, t)$, where k is the current number of the first position in C_i, d is the current number of the second position in C_j, and $t = |i-j|$.

The relation between the erasure events at two positions can be determined uniquely by the type t. If we denote their intrinsic and extrinsic erasures by e^k_{int}, e^k_{ext}, e^d_{int} and e^d_{ext} correspondingly, the following events can be distinguished:

- Totally Erased (TE): Both extrinsic messages are erased, $e^k_{ext} \bigcap e^d_{ext}$, independently of the intrinsic messages
- Erased-Known (EK): The extrinsic message at position k is erased, the other extrinsic message is known, $e^k_{ext} \bigcap \bar{e}^d_{ext}$[5], independently of the intrinsic messages
- Known-Erased (KE): $e^d_{ext} \bigcap \bar{e}^k_{ext}$, independently of intrinsic messages
- Copy at position k (Ck): the intrinsic erasure at position d implies the extrinsic erasure at position k, $e^d_{int} \Rightarrow e^k_{ext}$
- Copy at position d (Cd): $e^k_{int} \Rightarrow e^d_{ext}$
- Totally Known (TK): Both extrinsic messages are known, $\bar{e}^k_{ext} \bigcap \bar{e}^d_{ext}$, independently of intrinsic messages

The probabilities of the erasure events at some iteration l depend on the type $\mathbf{t}$, channel erasure probability ϵ and average intrinsic erasure probabilities $x^{(l-1)}, \ldots, x^{(l-t-1)}$. For our purpose, we are interested in calculating the probabilities of events TE, EK, KE, Ck and Cd.

Consider the trellis of a convolutional base code being a concatenation of identical local codes and let each trellis section correspond to a local code. Consider the trellis sections carrying positions k and d, and all the sections between these two. This forms the chain of length of $t+1$ sections (see Fig.4), we enumerate them from 0 to t.

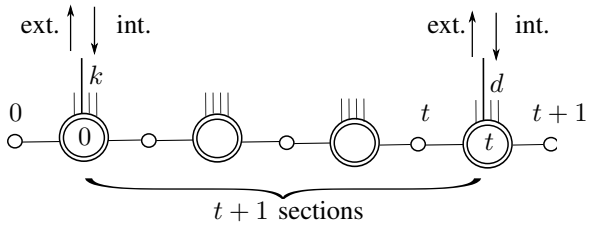

Fig. 4. Representation of the chain with $t+1$ trellis sections and of two positions k and d. Double circles represent trellis sections, small circles - states. Trellis section are enumerated from 0 to t. Sates are enumerated from 0 to $t+1$. Intrinsic messages at two positions of interest are given in blue, extrinsic ones - in red.

Let ϵ be the channel erasure probability and x be the intrinsic erasure probability. Recall from Section III-C that $\mathbf{f}_i$ and $\mathbf{b}_i$ are forward and backward states of the i-th trellis section. We begin the computation by considering the TE event.

1) TE Event: We compute $p_{TE}(\mathbf{t})$ directly as

$$\begin{aligned} p_{TE}(\mathbf{t}) &= \sum_{\mathbf{f}_0,\mathbf{f}_t,\mathbf{b}_1,\mathbf{b}_{t+1}} \mathbf{P}(e^k_{ext}, e^d_{ext}, \mathbf{f}_0, \mathbf{f}_t, \mathbf{b}_1, \mathbf{b}_{t+1}) \\ &= \sum_{\mathbf{f}_0,\mathbf{f}_t,\mathbf{b}_1,\mathbf{b}_{t+1}} \mathbf{P}(\mathbf{f}_0)\cdot\mathbf{P}(e^k_{ext}, \mathbf{f}_t|\mathbf{f}_0, \mathbf{b}_1) \\ &\quad \cdot\mathbf{P}(e^d_{ext}, \mathbf{b}_1|\mathbf{f}_t, \mathbf{b}_{t+1})\cdot\mathbf{P}(\mathbf{b}_{t+1}), \end{aligned}$$

where $\mathbf{f}_i \in S$ and $\mathbf{b}_j \in S$ for $i \in \{0,1,t\}$ and $j \in \{1,t,t+1\}$.[6] Now we need to find $\mathbf{P}(e^k_{ext}, \mathbf{f}_t|\mathbf{f}_0, \mathbf{b}_1)$ and $\mathbf{P}(e^d_{ext}, \mathbf{b}_1|\mathbf{f}_t, \mathbf{b}_{t+1})$. Both these quantities depend on values at positions in the chain of Fig. 4 (except the two positions under consideration). We make the following notation:

[4] i,e, positions connected to variable nodes of degrees > 1

[5] We denote by $\bar{u}$ the event complementary to u.

[6] Recall that S is the set of possible state vectors under all-zero codeword assumption

Notation Let $\mathbf{x}$ (or $\mathbf{e}$) denote the set of positions in the chain of degree higher than 1 (or of degree 1), excepting the two positions under consideration.

Note that if a local code contains s positions of degrees greater than 1 and r positions of degree 1, then the length of $\mathbf{x}$ is $K = s(t+1) - 2$ and the length of $\mathbf{e}$ is $M = r(t+1)$. By conditioning on the values of $\mathbf{x}$ and $\mathbf{e}$, we obtain that

$$\begin{aligned} p_{TE}(\mathbf{t}) &= \sum_{\mathbf{f}_0,\mathbf{b}_{t+1}} \mathbf{P}(\mathbf{f}_0)\cdot\mathbf{P}(\mathbf{b}_{t+1}) \sum_{\mathbf{x},\mathbf{e}} \mathbf{P}(\mathbf{x},\mathbf{e}) \\ &\quad \cdot\mathbf{P}(e^k_{ext}|\mathbf{x},\mathbf{e},\mathbf{f}_0,\mathbf{b}_1)\cdot\mathbf{P}(e^d_{ext}|\mathbf{x},\mathbf{e},\mathbf{f}_t,\mathbf{b}_{t+1}), \end{aligned}$$

where $\mathbf{P}(e^k_{ext}|\mathbf{x},\mathbf{e},\mathbf{f}_0,\mathbf{b}_1)$ and $\mathbf{P}(e^d_{ext}|\mathbf{x},\mathbf{e},\mathbf{f}_t,\mathbf{b}_{t+1})$ are conditionally independent. Thus,

$$\begin{aligned} p_{TE}(\mathbf{t}) &= \sum_{\mathbf{x}\in\{0,1\}^K} \sum_{\mathbf{e}\in\{0,1\}^M} \epsilon^{wt(\mathbf{e})}(1-\epsilon)^{1-wt(\mathbf{e})} x^{wt(\mathbf{x})} \\ &\quad (1-x)^{1-wt(\mathbf{x})}\cdot[\mathbf{f}^T.\left(W_{TE} * R_{TE}\right).\mathbf{b}], \end{aligned} \quad (8)$$

where $wt(\cdot)$ is the Hamming weight of a binary vector, and W_{TE} and R_{TE} are matrices with following elements:

$$W_{TE}[u,v] = \mathbf{P}(e^d_{ext}|\mathbf{f}=\mathbf{s}_u,\mathbf{x},\mathbf{e},\mathbf{b}=\mathbf{s}_v,x) \quad (9)$$

$$R_{TE}[u,v] = \mathbf{P}(e^k_{ext}|\mathbf{f}=\mathbf{s}_u,\mathbf{x},\mathbf{e},\mathbf{b}=\mathbf{s}_v,x) \quad (10)$$

for $u,v = 1,\ldots,p$. In other words, the elements of W_{TE} (R_{TE}) are conditional probabilities of the erasure event e^d_{ext} (e^k_{ext}) given the end states of the trellis chain, configurations $\mathbf{e}$ and $\mathbf{x}$ for positions of degree 1 and of higher degrees in the chain, and the intrinsic erasure probability $\mathbf{P}(e^k_{int}) = x$ ($\mathbf{P}(e^d_{int}) = x$).

There is another way, perhaps more intuitive, to obtain $p_{TE}(\mathbf{t})$: by considering all the erasure configurations at the trellis chain which give us this erasure event. Notice that erasures at two different trellis positions cannot be compensated by the decoding of the convolutional code, if there exist four codewords, equal to $(0,0)$, $(0,1)$, $(1,0)$ and $(1,1)$ at these two positions respectively, which are compatible with the received erasure configuration. By calculating the probability of this event conditioned on the received erasure configuration and by summing over all the possible erasure configurations, we obtain again (8).

Let us compute matrices W_{TE} and R_{TE}. First note that $\mathbf{x}$ and $\mathbf{e}$ can be seen as sets of subvectors, $\mathbf{x} = (\mathbf{x}_0,\ldots,\mathbf{x}_t)$ and $\mathbf{e} = (\mathbf{e}_0,\ldots,\mathbf{e}_t)$, where $\mathbf{x}_i$ ($\mathbf{e}_i$) is the set of values at positions of degree > 1 (equal to 1) of the i-th local code. Denote by $F^{\mathbf{x}_i,\mathbf{e}_i}$ the forward transition matrix for the i-th section given $\mathbf{x}_i$ and $\mathbf{e}_i$, $i = 1\ldots t-1$. $F^{\mathbf{x}_i,\mathbf{e}_i}$ is a $0-1$ matrix. Moreover, notice that the backward transition matrix $B^{\mathbf{x}_i,\mathbf{e}_i} = (F^{\mathbf{x}_i,\mathbf{e}_i})^T$. We also define $F^{\mathbf{x}_0,\mathbf{e}_0}(x)$ (or $B^{\mathbf{x}_t,\mathbf{e}_t}(x)$) to be the forward (or backward) transition-probability matrix for the section 0 (or t), given corresponding subvectors of $\mathbf{x}$ and $\mathbf{e}$, and the erasure intrinsic probability for the position k (or d) equal to x. Finally, let $T^{\mathbf{x},\mathbf{e}}_d$ be a matrix, the $[u,v]$-th element of which is

$$T^{\mathbf{x},\mathbf{e}}_d[u,v] = \mathbf{P}(e^d_{ext}|\mathbf{f}[u],\mathbf{b}[v],\mathbf{x},\mathbf{e}).$$

Then, for $u,v = 1,\ldots,p$,

$$\begin{aligned} W_{TE} &= F^{\mathbf{x}_0,\mathbf{e}_0}(x)\left(\times\prod_{i=1}^{t-1} F^{\mathbf{x}_i,\mathbf{e}_i}\right)\times T^{\mathbf{x}_t,\mathbf{e}_t}_d, \\ R_{TE} &= T^{\mathbf{x}_0,\mathbf{e}_0}_k \times\left(\prod_{i=1}^{t-1} F^{\mathbf{x}_i,\mathbf{e}_i}\right)\times (B^{x,\mathbf{x}_t,\mathbf{e}_t})^T. \end{aligned}$$

2) EK and KE Events: It can be shown that, similarly to (8), the expressions for $p_{EK}(\mathbf{t})$ and $p_{KE}(\mathbf{t})$ are given by

$$\begin{aligned} p_{EK}(\mathbf{t}) &= \sum_{\mathbf{x}\in\{0,1\}^K} \sum_{\mathbf{e}\in\{0,1\}^M} \epsilon^{wt(\mathbf{e})}x^{wt(\mathbf{x})}(1-\epsilon)^{1-wt(\mathbf{e})} \\ &\quad \cdot(1-x)^{1-wt(\mathbf{x})}\cdot[\mathbf{f}^T.\left(W_{EK} * R_{EK}\right).\mathbf{b}], \\ p_{KE}(\mathbf{t}) &= \sum_{\mathbf{x}\in\{0,1\}^K} \sum_{\mathbf{e}\in\{0,1\}^M} \epsilon^{wt(\mathbf{e})}x^{wt(\mathbf{x})}(1-\epsilon)^{1-wt(\mathbf{e})} \\ &\quad \cdot(1-x)^{1-wt(\mathbf{x})}\cdot[\mathbf{f}^T.\left(W_{KE} * R_{KE}\right).\mathbf{b}], \end{aligned}$$

where matrices W_{EK} and R_{EK} have elements

$$W_{EK}[u,v] = \mathbf{P}(\bar{e}^d_{ext}|\mathbf{f}=\mathbf{s}_u,\mathbf{x},\mathbf{e},\mathbf{b}=\mathbf{s}_v,x), \quad (11)$$

$$R_{EK}[u,v] = \mathbf{P}(e^k_{ext}|\mathbf{f}=\mathbf{s}_u,\mathbf{x},\mathbf{e},\mathbf{b}=\mathbf{s}_v,1-x), \quad (12)$$

while for the KE event we have

$$W_{KE}[u,v] = \mathbf{P}(e^d_{ext}|\mathbf{f}=\mathbf{s}_u,\mathbf{x},\mathbf{e},\mathbf{b}=\mathbf{s}_v,1-x), \quad (13)$$

$$R_{KE}[u,v] = \mathbf{P}(\bar{e}^k_{ext}|\mathbf{f}=\mathbf{s}_u,\mathbf{x},\mathbf{e},\mathbf{b}=\mathbf{s}_v,x). \quad (14)$$

In both cases $u,v = 1,\ldots,p$.

3) Copy Events: Now we only need to compute the probabilities of copy events. We use the following lemma:

Lemma 2:

$$\begin{aligned} p_{Ck}(\mathbf{t}) &= \mathbf{P}(e^k_{ext}|e^d_{int}) - \mathbf{P}(e^k_{ext}|\bar{e}^d_{int}) \\ p_{Cd}(\mathbf{t}) &= \mathbf{P}(e^d_{ext}|e^k_{int}) - \mathbf{P}(e^d_{ext}|\bar{e}^k_{int}). \end{aligned}$$

Proof: It is easy to see that

$$\begin{aligned} \mathbf{P}(e^k_{ext}) &= \mathbf{P}(e^k_{ext}|\bar{e}^d_{int}) + \mathbf{P}(e^d_{int})[\mathbf{P}(e^k_{ext}|e^d_{int}) \\ &\quad -\mathbf{P}(e^k_{ext}|\bar{e}^d_{int})] \\ &= \mathbf{P}(e^k_{ext}|e^k_{int} \not\Rightarrow e^d_{ext}) + \mathbf{P}(e^d_{int})p_{Ck}(h), \end{aligned}$$

which directly gives us the expression for $p_{Ck}(\mathbf{t})$. The case of $p_{Cd}(\mathbf{t})$ is treated similarly and we omit the calculation. ■

Developing the expression for $p_{Ck}(\mathbf{t})$ for $t > 0$, we obtain

$$\begin{aligned} p_{Ck}(\mathbf{t}) &= \mathbf{f}^T.T_k.(B^T)^{t-1}.(B_d(1) - B_d(0)^T.\mathbf{b}, \\ p_{Cd}(\mathbf{t}) &= \mathbf{f}^T.T_d.(B^T)^{t-1}.(B_k(1) - B_k(0)^T.\mathbf{b}, \end{aligned}$$

where we denote by $B_d(z)$ the backward transition-probability matrix B of the trellis section, for which the erasure probability of the position d is z.

4) Quantities p_C, p_{TE} and p_{ELR}: Having defined the erasure events for two positions at distance t in a convolutional code, we are ready to define the following quantities

$$\begin{aligned} p_C = \frac{1}{s}\Bigg(&\sum_{1\le k\le s}\sum_{1\le d\le s, d\ne k} p_{Ck/Cd}(0,k,d) \\ &+ \sum_{1\le d\le s}\sum_{t=1}^{\infty}[p_{Ck}(k,d,t) + p_{Cd}(k,d,t)]\Bigg), \end{aligned} \quad (15)$$

$$p_{TE} = \frac{1}{s}\{ \sum_{1\le k\le s} \sum_{1\le d\le s, d\neq k} p_{TE}(0,k,d) + 2 \sum_{1\le d\le s} \sum_{t=1}^{\infty} p_{TE}(k,d,t)\}, \quad (16)$$

$$p_{ELR} = \frac{1}{s}\{ \sum_{1\le k\le s} \sum_{1\le d\le s, d\neq k} p_{EK/KE}(0,k,d) + \sum_{1\le d\le s} \sum_{t=1}^{\infty} [e^d_{ext}(k,d,t) + p_{KE}(k,d,t)]\}, \quad (17)$$

where $p_{EK/KE}(0,k,d)$ ($p_{Ck/Cd}(0,k,d)$) can define either the event EK or KE (either the event Ck or Cd), depending on how positions k and d are located in the same trellis section.

We also need to define one more quantity, which will be useful further in the calculation.

Definition 5: Let $E_{\mathbf{t}}$ be the event e^k_{ext} for the k-th position of the j-th convolutional node at iteration $l-j+1$ irrespective of the value of the position d at distance t at the previous iteration.

By definition, the probability of E_j is

$$\mathbf{P}(E_{\mathbf{t}}) \quad = \quad p_{TE}^{(l-j+1)}(\mathbf{t}) + p^d_{EK}(l-j+1)(\mathbf{t}),$$

if the position k precedes the position d and

$$\mathbf{P}(E_{\mathbf{t}}) \quad = \quad p_{TE}^{(l-j+1)}(\mathbf{t}) + p_{KE}^{(l-j+1)}(\mathbf{t}),$$

if the position d precedes the position k in the trellis.

B. Computing ν

We begin the second part of our calculation in which we calculate the variance ν itself, by using the quantities defined in the previous part. The proof consists of computing the expression $\nu^{(l)}$ for finite number of iterations l (at infinite length) and taking the limit $l \to \infty$ afterwards and is similar to the one presented for LDPC codes.

1) Two Contributions of ν: Consider a turbo-like code and suppose the transmission takes place over BEC with the erasure probability ϵ. Perform l iterations of the iterative decoding. Take at random an edge in the corresponding Tanner graph and consider a message going from the variable node to a convolutional node. As the code is assumed to be of infinite length, then the convolutional code is of infinite length as well. So, any two distinct positions of it are located far from each other and the extrinsic probabilities calculated for these positions are independent as if they came from two separate convolutional codes. Then the computation tree of the edge under consideration, which consists of the variable and the convolutional nodes, can be assumed to be tree-like.

We say that an edge in the computation tree is at depth i from the root edge if the path connecting them contains i convolutional nodes. Let there are N_i edges at the depth i correlated with the root message and denote by $e_i^{(l)}$ the event that the message sent from a variable node to a convolutional node at the depth i is erased at the iteration l. Then we can compute $\nu^{(l)}$ as follows

$$\nu^{(l)} = \sum_{i=0}^{\infty} N_i(\mathbf{P}(e_1^{(l)}, e_i^{(l)}) - \mathbf{P}(e_1^{(l)})\mathbf{P}(e_i^{(l)})) \quad (18)$$

To compute the i-th term of the sum, consider the path between the root edge and an edge at depth i. Figure 5 represents such a path. We label the variable nodes (circles) from 0 to i and the convolutional nodes (blocks) from $\hat{1}$ to $\hat{i}$. Denote by $e^{(l)}_{a\to b}$ the event that the message going from the node a to the node b is erased at the iteration l.

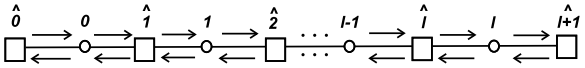

Fig. 5. Path of depth i in the computation tree. Circles represent variable nodes, blocks - convolutional nodes.

We are interested in the correlation between messages of type $a \to \hat{b}$. As there are messages going at the same and at the opposite direction and will contribute differently to (18), we represent $\nu^{(l)}$ as the sum of two contributions, from messages directed in the same and opposite way respectively. We denote these contributions as $\nu^{(l)}_{\rightarrow}$ and $\nu^{(l)}_{\leftrightarrow}$.

2) Variance of Erased Messages Directed in the Same Way: Consider two messages going out from variable nodes at distance i from each other at some iteration l and directed in the same way as it is shown in Fig. 6. As before, the blocks represent convolutional nodes and circles represent variable nodes. We depict by red arrows the messages with the corresponding iteration numbers on which the message going out from the i-th variable node is dependent.

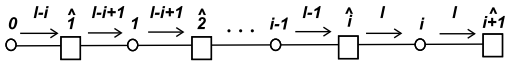

Fig. 6. Illustration to the computation of $\nu^{(l)}_{\rightarrow}$.

Evaluating conditional probabilities along the chain, we can show that

$$\mathbf{P}(e^{(l)}_{i\to\widehat{i+1}}, e^{(l)}_{0\to\widehat{1}}) - \mathbf{P}(e^{(l)}_{i\to\widehat{i+1}})\mathbf{P}(e^{(l)}_{0\to\widehat{1}}) = (1-x^{(l-i+1)})x^{(l-i)} \prod_{g=1}^{i} [p^{(l-i+g-1)}_{Ck/Cd}(\mathbf{t}_g)\cdot\epsilon \sum_{c=2}^{i_{max}} \tilde{\lambda}_c \left(y^{(l-i+g)}\right)^{c-2}],$$

where the triplets $\mathbf{t}_g$ are uniquely defined by the edge $0 \to \hat{1}$ to the edge $i \to i \hat{+} 1$ in the computation tree. We thus calculate

$$\nu^{(l)}_{\rightarrow} = (1-x^{(l-i+1)})x^{(l-i)} \sum_{i=1}^{l} \prod_{g=1}^{i} \epsilon\tilde{\lambda}'(y^{(l-i+g)})p_C^{(l-i+g-1)}.$$

For $\epsilon > \epsilon^\star$ and $l \to \infty$, we obtain that

$$\nu_{\rightarrow} \quad = \quad x(1-x)\frac{\epsilon\tilde{\lambda}'(y)p_C}{1-\epsilon\tilde{\lambda}'(y)p_C}.$$

3) Variance of Erased Messages Directed Oppositely: The computation of $\nu_{\leftrightarrow}$ is more involved that the one of $\nu_{\rightarrow}$. Consider two messages going out from variable nodes at distance i from each other at some iteration l and directed in the opposite way (see Fig. 7). To make the computation we consider the messages at the previous iterations on which the aforementioned messages are dependent and we study the relations between them.

Let $U_{\mathbf{t}_j}$ denotes the event that the erasure $e^{(l)}_{0\to\widehat{0}}$ was generated by the erased message e^k_{ext} of the j-th convolutional

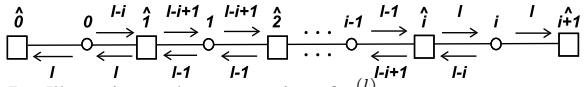

Fig. 7. Illustration to the computation of $\nu_{\leftrightarrow}^{(l)}$.

node happened at iteration $l-j+1$. The probability of $U_{\mathbf{t}_j}$ is then expressed as

$$\mathbf{P}(U_{\mathbf{t}_j}) \quad = \quad \mathbf{P}(E_{\mathbf{t}_j})\Pi_{g=1}^{j-1}[p_{Ck/Cd}^{(l-g)}(\mathbf{t}_g)\cdot\cdot\epsilon\sum_{c=2}^{i_{max}}\tilde{\lambda}_c\left(y^{(l-g)}\right)^{c-2}],$$

where the triplets $\mathbf{t}_g$ are uniquely defined by the edge $0\rightarrow\hat{1}$ to the edge $i\rightarrow i+1$ in the computation tree. By evaluating the conditional probabilities for the chain of Fig. 7, we obtain

$$\mathbf{P}(e_{0\rightarrow\widehat{0}}^{(l)}, e_{i\rightarrow\widehat{i+1}}^{(l)}) - \mathbf{P}(e_{0\rightarrow\widehat{0}}^{(l)})\mathbf{P}(e_{i\rightarrow\widehat{i+1}}^{(l)}) =$$
$$\mathbf{P}(e_{0\rightarrow\widehat{0}}^{(l)})\sum_{j=1}^{l}\mathbf{P}(U_{\mathbf{t}_j})(\mathbf{P}(e_{i\rightarrow\widehat{i+1}}^{(l)}|e_{0\rightarrow\widehat{0}}^{(l)}, U_{\mathbf{t}_j}) - \quad \mathbf{P}(e_{i\rightarrow\widehat{i+1}}^{(l)})),$$

with

$$\mathbf{P}(e_{i\rightarrow\widehat{i+1}}^{(l)}|e_{0\rightarrow\widehat{0}}^{(l)}, U_{\mathbf{t}_j}) - \mathbf{P}(e_{i\rightarrow\widehat{i+1}}^{(l)}) = (\epsilon\sum_{c=2}^{i_{max}}\tilde{\lambda}_c\left(y^{(l)}\right)^{c-2})$$
$$\cdot\Pi_{h=1}^{i-j}\left(\epsilon\sum_{c=2}^{i_{max}}\tilde{\lambda}_c\left(y^{(l-h)}\right)^{c-2}\cdot p_{Ck/Cd}^{(l-h)}(\mathbf{t}_{i-h+1})\right)$$
$$\cdot\left(\mathbf{P}(e_{\widehat{j}\rightarrow j}^{(l+j-i)}|e_{0\rightarrow\widehat{0}}^{(l)}, U_{\mathbf{t}_j}) - \mathbf{P}(e_{\widehat{j}\rightarrow j}^{(l+j-i)})\right).$$

Here there are two case to distinguish: $j\leq i-j$ and $j>i-j$. For more details see Appendix. At the fixed point, i.e. for $\epsilon>\epsilon^\star$ and $l\rightarrow\infty$, the expression simplifies and is equal to

$$\mathbf{P}(e_{\widehat{j}\rightarrow j}|e_{0\rightarrow\widehat{0}}, E_{\mathbf{t}_j}) - \mathbf{P}(e_{\widehat{j}\rightarrow j}) = \frac{p_{TE}(\mathbf{t}_j) - y\mathbf{P}(E_{\mathbf{t}_j})}{\mathbf{P}(E_{\mathbf{t}_j})}.$$

Therefore, $\nu_{\leftrightarrow}$ is calculated as

$$\nu_{\leftrightarrow} \quad = \quad \sum_{i=1}^{\infty} i\frac{\epsilon\tilde{\lambda}'(y)}{p_C}\left(\epsilon\tilde{\lambda}'(y)p_C\right)^i\gamma = \frac{(\epsilon\tilde{\lambda}'(y))^2\gamma}{(1-\epsilon\tilde{\lambda}'(y)p_C)^2}$$

with

$$\gamma = (1-y)p_{TE} - yp_{ELR}, \tag{19}$$

which is the expression stated in (7).

VII. Numerical Estimation of Shift Parameter β

In this section we give a simple but efficient procedure of numerical estimation of the shift parameter β for a given turbo-like ensemble. This estimation requires the simulation of the ensemble performance around the block erasure probability $1/2$.

Consider a turbo-like code ensemble of length N, having n bits of degrees >1, and assume the transmission over the binary erasure channel. We define the threshold of this ensemble:

Notation [[12]]: The *threshold* $\epsilon(N)$ of a code ensemble of length N, having n bits of degrees >1, is the smallest erasure probability such that the average erasure probability of the ensemble is at least one half, i.e.

$$\epsilon(N) = \inf\{\epsilon \text{ s.t. } P_B \geq 1/2\}.$$

By conjecturing the refined scaling law, we conjecture that

$$\epsilon(N) = \epsilon^\star - \beta n^{-2/3}.$$

Notice that $\epsilon(N)$ can be estimated numerically by simulating the block erasure probability of a chosen ensemble around the value $1/2$, and the value of asymptotic iterative threshold $\epsilon^\star$ is given by the density evolution equation. Therefore,

$$\beta = n^{2/3}(\epsilon^\star - \epsilon(N)).$$

Remark Describing the numerical estimation of β, we assumed that the given code ensemble has only one critical point. If the code ensemble has several but distinguishable critical points, it is easy to estimate two thresholds $\epsilon_1(N)$ and $\epsilon_2(N)$ from the numerical data: errors coming from different critical points are distinguishable by their weight.

VIII. Examples

In this section we apply our calculation to different turbo-like ensembles to obtain their scaling parameter and to estimate the slope of their average error probability curves in the waterfall region. We also estimate their shift parameters β's using the approach explained in the previous section. Then we compare the obtained approximations with the numerical results.

A. Repeat-Accumulate codes

Let us apply the aforementioned results to the case of right-regular RA codes with the degree of check nodes equal to d.

For simplicity we define the following quantities:

$$p_\oplus = (1-x)^d, \qquad s = \frac{\epsilon(1-p_\oplus)}{1-\epsilon p_\oplus}, \qquad a = \frac{1-s}{1-x},$$
$$y = 1-(1-s)ap_\oplus;$$

then the fixed point density evolution equation can be written as

$$\epsilon\tilde{\lambda}(y) - x = 0.$$

Further, we find

$$p_C \quad = \quad a^2p_\oplus\left(\frac{2d}{1-\epsilon p_\oplus} - d - 1\right).$$

and

$$\gamma = (1-y)\left[(d-1)(1-a^2p_\oplus)\right.$$
$$+\frac{2d\epsilon p_\oplus}{1-\epsilon p_\oplus}\left(y^2 + 2a(1-s) - a^2p_\oplus - a\frac{(\epsilon-s)^2}{\epsilon^2}\right)$$
$$\left.+\frac{2d\epsilon p_\oplus}{(1-\epsilon p_\oplus)^2}a(1-p_\oplus)\left(y + a\frac{\epsilon-s}{\epsilon}\right)\right].$$

The three last equations are everything that we need to find the terms in the expression (4) for α.[7]

As an example, consider the $(3,3)$ RA ensemble that is of rate $1/2$ and for which we have $\epsilon^\star = 0.44478$, $y^\star = 0.6322$ and $x^\star = 0.1778$. The scaling parameter α in this case is equal to 0.44436. The shift parameter β, estimated numerically, is equal to 0.45.

[7] for more details about obtaining p_C and γ see [4]

Figure 8 compares simulation results (solid lines) with the finite-length estimations for information lengths 1024, 2048, 4096 and 8192, obtained with the basic (dotted lines and filled signs) and the refined (unfilled signs) scaling approach. Notice that the estimated curves have the same slope as the simulated ones and their refined scaling law seems to approximate very well the behaviour of the word error probability.

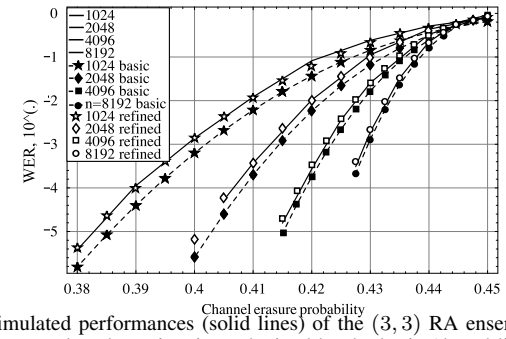

Fig. 8. Simulated performances (solid lines) of the $(3,3)$ RA ensemble on the BEC compared to the estimations obtained by the basic (dotted lines and filled signs) and refined (unfilled signs) scaling approach.

B. Parallel Turbo Codes

In the general case of turbo codes, no simplifications can be made as with RA codes and we simply apply the calculation of Section V directly. Let us apply it to an example.

Consider the $(1, \frac{1+D^2}{1+D+D^2}, \frac{1+D^2}{1+D+D^2})$ turbo ensemble of rate $1/3$, for which $\epsilon^\star = 0.642831$, $y^\star = 0.6240$ and $x^\star = 0.4011$. Using Lemma 1, we calculate the scaling parameter α, which is equal to 0.331. By numerical estimation, we also estimate that $\beta = 0.17$ Figure 9 compares simulation results with the finite-length estimation for information lengths 512, 1024 and 2048. Also, we obtain a very good approximation of the performance estimation.

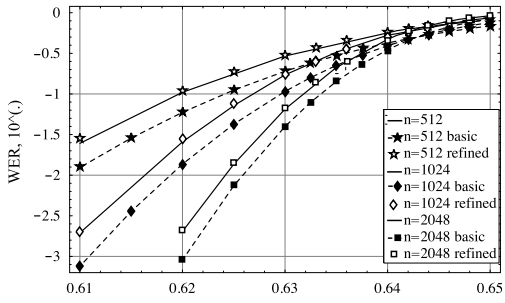

Fig. 9. Simulated performances (solid lines) of the $(1, \frac{1+D^2}{1+D+D^2}, \frac{1+D^2}{1+D+D^2})$ turbo ensemble on the BEC compared to the estimations obtained by the basic (dotted lines and filled signs) and (unfilled signs) refined scaling approach.

C. TLDPC Codes

As an example, take the TLDPC ensemble for which $a_i = b_i = 3$ for all $i \in [0, m-1]$[8] and the degree distribution $\lambda(x)$ of which is

$$\lambda(x) = 0.61441700231x + 0.02989470758x^7 + 0.007007531202x^8 + 0.2858599574x^{19}.$$

[8]Let us denote the ensemble as $(3,3)$ TLDPC

Notice that $\lambda_1 = 0$ and that the code ensemble has rate $1/2$.

Let us also define the following quantities:

$$p = 1-(1-x)^3, \quad q = 1-(1-x)^2$$
$$s_1 = \frac{p^2}{1-p(1-p)}, \qquad s_2 = \frac{p}{1-p(1-p)}$$

$$y = \frac{1}{2}[1 + s_2^2 - (1-s_1)^2 + q(1 - s_2^2 + (1-s_1)^2)];$$

then the fixed point density evolution equation can be written as

$$\epsilon\lambda(y) - x = 0.$$

and it has two critical points, $(\epsilon_1^\star, x_1^\star) = (0.483, 0.3865)$ and $(\epsilon_2^\star, x_2^\star) =$
$= (0.482975, 0.19075)$. As in the case of Repeat-Accumulate codes, the quantities p_C and γ can be computed explicitly, and we can find two scaling parameters, α_1 and α_2, corresponding to two critical points. We obtain that $\alpha_1 = 0.5192$, $\alpha_2 = 0.8588$. Numerically, we estimate that $\beta_1 = 1.18$ and $\beta_2 = 1.27$. To obtain the total estimation of the word error probability for $(3,3)$ TLDPc codes, we have that

$$P_B = P_B|_{\alpha=\alpha_1,\beta=\beta_1} + (1 - P_B|_{\alpha=\alpha_1,\beta=\beta_1})P_B|_{\alpha=\alpha_2,\beta=\beta_2}.$$

In Figure 10 we compare the obtained estimations with the simulated results.

IX. Conclusion

In this paper we have conjectured the basic and refined scaling laws for irregular turbo-like code ensembles, assuming that the transmission takes place over the BEC. We have also defined the form of their scaling parameter α by using the techniques already applied for irregular LDPC codes. By calculating α in the case of Repeat-Accumulate codes, of parallel turbo codes and of TLDPC codes and by estimating the shift parameter β numerically in these three cases, we have obtained the finite-length approximations of the average word error probability for the aforementioned code ensembles, which match very well with the simulated results even for moderate code lengths.

Our technique is based on the computation of different erasure events in the underlying convolutional-like codes of turbo-like ensembles, and it is the generalization of the approach that was presented in [2] in the case of LDPC ensembles.

In some cases the explicit calculation of the probabilities of erasure events becomes challenging. For example, when the number of trellis states of the base code is greater than 4, the number of possible state vectors increases rapidly, which makes the calculation quite heavy. These probabilities however can be estimated numerically only by performing the ML decoding of the underlying base code and should be done only once for a given code ensemble.

Finally, notice that we have only considered one particular (yet the most common) case when all local codes in the base code structure are identical. With a bit of work the result can be extended to a general case with different local codes.

X. Acknowledgment

The author would like to thank Rudiger Urbanke and Jean-Pierre Tillich for very helpful comments and suggestions.

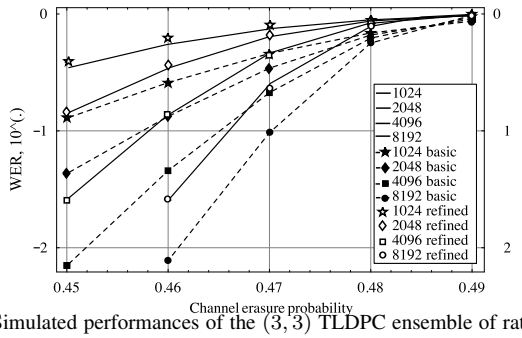

Fig. 10. Simulated performances of the $(3,3)$ TLDPC ensemble of rate $1/2$ on the BEC compared to the estimations obtained by the basic and refined scaling approach.

REFERENCES

[1] A. Amraoui, A. Montanari, T. Richardson, and R. Urbanke. Finite-length scaling for iteratively decoded LDPC ensembles. *IEEE Trans. on Information Theory*, 55(2):473–498, February 2009.

[2] A. Amraoui, A. Montanari, and R. Urbanke. How to find good finite-length codes: from art towards science. *European Transactions on Communications*, 18(5):491–508, August 2007.

[3] I. Andriyanova. Finite-length scaling of parallel turbo codes on the BEC. In *ISIT*, pages 1893 –1897, July 2008.

[4] I. Andriyanova. Finite-length scaling of repeat-accumulate codes on the BEC. In *IZS*, pages 64–67, March 2008.

[5] I. Andriyanova, J.P. Tillich, and J.C. Carlach. Asymptotically good codes with high iterative decoding performances. In *ISIT*, pages 850 –854, September 2005.

[6] I. Andriyanova, J.P. Tillich, and J.C. Carlach. A new family of asymptotically good codes with high iterative decoding performances. In *ICC*, pages 1177 –1182, June 2006.

[7] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimazing symbol error rate. *IEEE Trans. on Information Theory*, 20:284–287, March 1974.

[8] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding. In *ICC'93*, pages 1064–1070, Genève, Switzerland, May 1993.

[9] E. Dembo and A. Montanari. Finite-size scaling for the core of large random hypergraphs. to appear in *Ann. Appl. Probab.*, 2008.

[10] C. Di, D. Proietti, E. Telatar, and R. Urbanke. Finite-length analysis of low-density parity-check codes on the binary erasure channel. *IEEE Trans. on Information Theory*, 48(6):1569–1579, June 2002.

[11] D. Divsalar, H. Jin, and R. J. McEliece. Coding theorems for 'turbo-like' codes. In *Proc. 36th Allerton Conf. on Coomunication, Control, and Computing.*, pages 201–210, Allerton, Illinois, September 1998.

[12] J. Ezri, A. Montanari, S. Oh, and R. Urbanke. Computing the threshold shift for general channels. In *ISIT'08*, pages 1448–1452, Toronto, Canada, June 2008.

[13] J. Ezri, A. Montanari, and R. Urbanke. Finite-length scaling for Gallager A. In *Proc. 44th Annual Allerton Conference on Communication, Control and Computing*, Illinois, September 2006.

[14] J. Ezri, A. Montanari, and R. Urbanke. A generalization of the finite-length scaling approach beyond the BEC. In *Proc. ISIT'2007*, Nice, France, June 2007.

[15] M. E. Fisher. *In Critical Phenomena*. M.S. Green, Academic, New York, 1971.

[16] K. Fu and A. Anastasopoulos. Stopping set enumerator approximations for finite-length protograph LDPC codes. In *Proc. ISIT'2007*, Nice, France, June 2007.

[17] H. Jin, A. Khandekar, and R. J. McEliece. Irregular repeat-accumulate codes. In *Proc. 2nd International Symposium on Turbo Coding*, pages 1–8, 2000.

[18] R. Johannesson. and K. Zigangirov. *Fondamentals of Convolutional Coding*. IEEE Press, 1999.

[19] S. Johnson. Finite-length repeat-accumulate codes in the binary erasure channel. In *Proc. Asia-Pacific Conference on Communications*, pages 217–221, Perth, Australia, 2005.

[20] J.W. Lee, R. Urbanke, and R.E. Blahut. On the performance of turbo codes over the binary erasure channel. *IEEE Communications Letters*, 11(1):67–69, 2007.

[21] J.W. Lee, R. Urbanke, and R.E. Blahut. Turbo codes in binary erasure channel. *IEEE Transansactions on Information Theory*, 54(4):1765–1773, 2008.

[22] W.K.Raymond Leung, Guosen Yue, Li Ping, and Xiaodong Wang. Concatenated zigzag Hadamard codes. *IEEE Trans. Inform. Theory*, 52(4):1711–1723, April 2006.

[23] C. Measson. *Conservation Laws for Coding*. PhD thesis, EPFL, Switzerland, 2006.

[24] O. Milenkovic, E. Soljanin, and P. Whiting. Asymptotic spectra of trapping sets in regular and irregular LDPC code ensembles. *IEEE Trans. on Information Theory*, 53(1):39–55, January 2007.

[25] H. Pfister. Finite-length analysis of a capacity-achieving ensemble for the binary erasure channel. In *Proc. IEEE Inform. Theory Workshop*, Rotorua, New Zealand, September 2005.

[26] T. Richardson and R. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Trans. Inform. Theory*, 47:599–618, February 2001.

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

[28] E. Rosnes and O. Ytrehus. Turbo decoding of the binary erasure channel: Finite-length analysis and turbo stopping sets. *IEEE Trans. Inform. Theory*, 53(11):4059–4075, November 2007.

APPENDIX

First, notice that

$$\mathbf{P}(e^{(l)}_{i\to\widehat{i+1}}|e^{(l)}_{0\to\widehat{0}},U_{\mathbf{t}_j})-\mathbf{P}(e^{(l)}_{i\to\widehat{i+1}})=\mathbf{P}(e^{(l)}_{i\to\widehat{i+1}}|e^{(l)}_{0\to\widehat{0}},E_{\mathbf{t}_j})-\mathbf{P}(e^{(l)}_{i\to\widehat{i+1}}).$$

and, moreover,

$$\mathbf{P}(e^{(l)}_{i\to\widehat{i+1}}|e^{(l)}_{0\to\widehat{0}},E_{\mathbf{t}_j})=\mathbf{P}(e^{(l)}_{i\to\widehat{i+1}}|E_{\mathbf{t}_j}).$$

To find the last quantity we need to distinguish two different cases: when $j\le i-j$ and when $j>i-j$.

1) $j\le i-j$; it is easy to calculate that

$$\mathbf{P}\left(e^{l+j-i}_{j\to\widehat{j+1}}|E_{\mathbf{t}_j}\right)=\epsilon\sum_{c=2}^{i_{max}}\tilde{\lambda}_c\left(y^{(l+j-1)}\right)^{c-2};$$

2) $j>i-j$; let $q^{(k)}=\epsilon\sum_{c=2}^{i_{max}}\tilde{\lambda}_c\left(y^{(k)}\right)^{c-2}$, then after some tedious calculations

$$\mathbf{P}\left(e^{l+j-i}_{j\to\widehat{j+1}}|E_{\mathbf{t}_j}\right)=q^{(l+j-i)}\left[\frac{1-p_C^{l+j-i-1}}{1-p_C^{l-j+1}}+\sum_{n=2}^{j-i/2}\frac{1-p_C^{l+j-i-n}}{1-p_C^{l-i+n}}\Pi_{k=1}^{n-1}\frac{q^{(l+j-i-k)}(p_C^{l+j-i-k}-p_C^{l-j+k})}{q^{(l-j+k)}(1-p_C^{l-j+k})}\right]$$

Fortunately, at the threshold $\epsilon^\star$ this quantity simplifies and is equal to $\epsilon^\star\sum_{c=2}^{i_{max}}\tilde{\lambda}_c\left(y^\star\right)^{c-2}$ in both cases.

Iryna Andriyanova received her PhD and MSc from the Communications&Electronics Department of National Superior School of Telecommunications (Telecom ParisTech), France, and BSc from the National Polytechnic University of Odessa, Ukraine. From 2007 to 2009 she was a postdoctoral researcher at the Federal Polytechnic School of Lausanne (EPFL), Switzerland. She is currently at the National Superior School of Electronics and its Applications (ENSEA), France. Her main research interests include coding and communication systems.