ISSN 0032-9460, Problems of Information Transmission, 2022, Vol. 58, No. 2, pp. 122–136. © Pleiades Publishing, Inc., 2022.
Russian Text © The Author(s), 2022, published in Problemy Peredachi Informatsii, 2022, Vol. 58, No. 2, pp. 24–40.

CODING THEORY

Theoretical and Experimental Upper and Lower Bounds on the Efficiency of Convolutional Codes in a Binary Symmetric Channel

A. A. Kurmukova[a,b,c,*], **F. I. Ivanov**[a,b,**], **and V. V. Zyablov**[a,***]

[a]*Kharkevich Institute for Information Transmission Problems, Russian Academy of Sciences, Moscow, Russia*
[b]*Higher School of Economics—National Research University, Moscow, Russia*
[c]*Skolkovo Institute of Science and Technology (Skoltech), Moscow, Russia*
e-mail: *Anastasiia.Kurmukova@skoltech.ru, **fivanov@hse.ru, ***zyablov@iitp.ru

Received December 12, 2021; revised March 8, 2022; accepted March 11, 2022

Abstract—We propose a new approach to the analytical estimation of the error burst probability, the probability of erroneous decoding, and the probability of error per bit for convolutional codes with Viterbi decoding in a binary symmetric channel (BSC). Upper and lower estimates of the probability of error per bit and of the erroneous decoding probability are based on active distances and the distance spectrum of active distances for a convolutional code. The estimates are derived for rate 1/2 convolutional codes, but they can also be generalized to any convolutional code with rate $1/n$. Calculation of the estimates described here has linear time complexity in the error burst minimal length if code distance properties are known. The computational complexity does not depend on the crossover probability of a BSC. Simulation results show that the considered estimates are rather tight, especially for small crossover probabilities.

Key words: convolutional codes, active distance, bit error rate, code trellis.

DOI: 10.1134/S003294602202003X

1. INTRODUCTION

Convolutional codes introduced by Elias in [1] have been studied for more than 60 years. Although convolutional codes are inferior in error correction performance to such codes as polar [2] or low-density parity-check (LDPC) codes [3], these codes can significantly reduce the bit-error rate (BER) for a given signal-to-noise ratio (SNR) [4]. Neither polar nor LDPC codes possess this property [4]. Apart from convolutional codes, only several classes of codes have this property, for instance, Hamming codes [5], some classes of LDPC codes with low-density generator matrices (LDGM) [6], repeated code constructions [7].

The property of BER reduction is very important for inner codes in any concatenated schemes. This is a major reason why convolutional codes are widely used in different concatenated schemes such as woven codes [8] and in other sequential and parallel constructions [9,10]. Also, the recursive systematic convolutional encoder has gained great popularity in turbo codes [11,12]. It also shows great performance in concatenation with other linear codes, for example, in a parallel scheme with a systematic polar code (SPC) with iterative decoding as in [13].

Thus, analytical methods for the error probability estimation for convolutional codes are an important direction of research. There are several works devoted to the theoretical analysis of the performance of convolutional codes. A classical approach to estimation of the error probability based on the analysis of generating functions was presented in [14]. For the binary symmetric channel, there is a tightened Meeberg bound [15]. In [16], simple Markov chain based models are used to evaluate the BER of convolutional codes for short constraint lengths. However, simulation results provided in this paper show that for a low input probability of error, the proposed estimation is rather imprecise. Some modifications of this technique were described in [17], where it was proposed to represent a particular convolutional code as a reduced Markov chain. In this case, Viterbi decoding [18] can be considered as a special transition process between states of a Markov chain. BER estimation for punctured convolutional codes was considered in [19]. Finally, in [20] it was shown how to estimate the BER for any convolutional code for both quantized AWGN (additive white Gaussian noise) and binary symmetric channels. However, the proposed technique is rather difficult in implementation and has high complexity for convolutional codes with large memory.

In this paper, we propose a new technique to estimate bit and frame error rate for convolutional codes with maximum likelihood decoding in a binary symmetric channel. The proposed approach is based on the distance spectrum of active distances and analytical estimates for the probability of error bursts with a given length [21, 22]. A frame error rate (FER) estimate for terminated convolutional codes was described in [23]. Here we propose simple theoretical low-complexity upper and lower bounds on the BER of convolutional codes.

2. CONVOLUTIONAL CODES

2.1. Convolutional Code with Recursive Encoder

In this paper, we for simplicity consider rate 1/2 convolutional codes with a recursive systematic encoder. At the same time, the obtained results can be generalized to convolutional codes with rate $1/n$, $n > 1$, $n \in \mathbb{N}$. All results here are valid for the BSC and the Viterbi decoder.

The generator matrix for rate 1/2 convolutional codes with a systematic encoder with feedback is given using the ratio of polynomials:

$$\boldsymbol{G}(D) = \begin{pmatrix} 1 & \dfrac{\boldsymbol{g}^{(2)}}{\boldsymbol{g}^{(1)}} \end{pmatrix}, \tag{1}$$

where the generator polynomials are

$$\boldsymbol{g}^{(\ell)}(D) = g_0^{(\ell)} + g_1^{(\ell)} D + g_2^{(\ell)} D^2 + \ldots + g_m^{(\ell)} D^m, \quad g_i^{(\ell)} \in \{0, 1\},$$

for $\ell = 1, 2$. The code memory is denoted here by m, and the code rate is 1/2. The feedback structure of the encoder corresponds to the denominator in the generator matrix. The generator polynomials must be co-prime and of the same degree. The encoder state at each time moment can be written either using m register bits or as a decimal number s: $0 \le s < 2^m$. The total number of different states is 2^m.

A codeword of a convolutional code can be written using a polynomial in indeterminate D (delay operator):

$$\boldsymbol{v}(D) = \boldsymbol{v}_0 + \boldsymbol{v}_1 D + \boldsymbol{v}_2 D^2 + \ldots, \quad \boldsymbol{v}_i \in \{0, 1\}^2, \quad i \in \mathbb{N} \cup \{0\},$$

where $(\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, \ldots)$ is a sequence of output tuples of size 2. A codeword of the convolutional code is semi-infinite and can also be uniquely defined with a sequence of input information bits and an initial encoder state.

It is convenient to use the code trellis to represent convolutional codewords and visualize the decoding process. The trellis is represented in the form of encoder states at successive time moments. A transition from state s_t at time moment t to state s_{t+1} at time moment $t+1$ corresponds to some input bit and some output tuple. The initial encoder state is usually assumed to be $t = 0$. Each

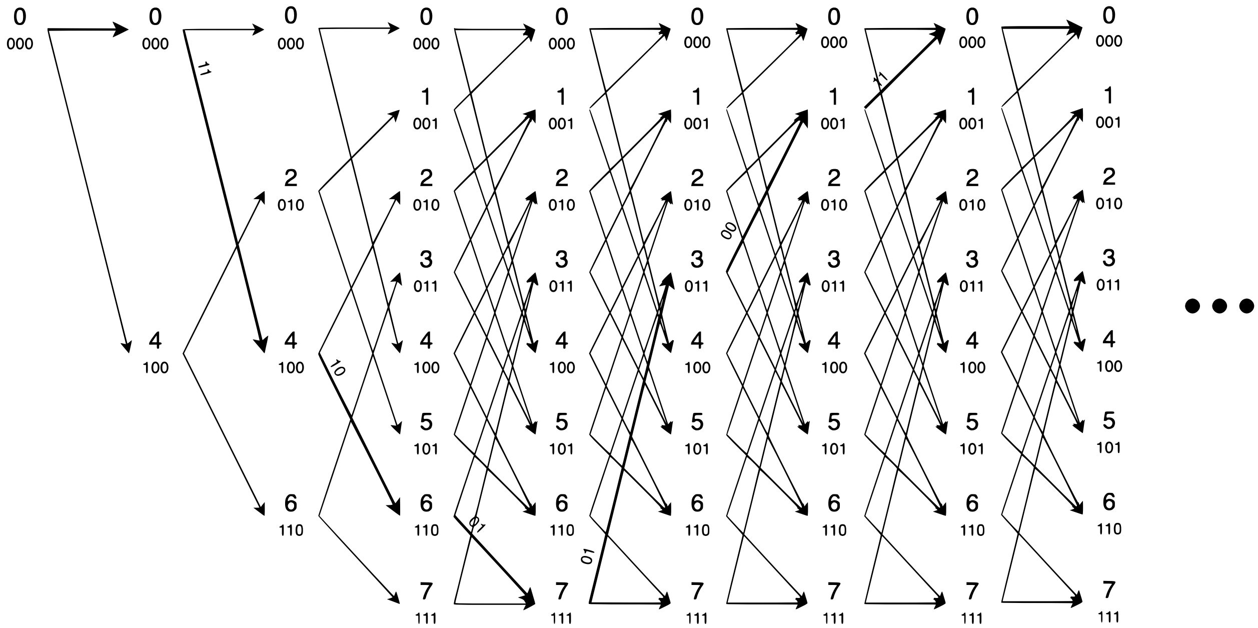

Fig. 1. Trellis for the $(13, 17)$ convolutional code with a recursive encoder.

codeword has a unique corresponding sequence of output tuples, a unique sequence of trellis states, and a unique sequence of input bits. Thus, each codeword can be written as a path in the trellis (a sequence of encoder states). The trellis representation is important for understanding distance properties of convolutional codes.

In Fig. 1 we provide an example for a recursive systematic convolutional encoder of memory $m = 3$ with generator polynomials $\boldsymbol{g}^{(1)} = 1 + D + D^3$ and $\boldsymbol{g}^{(2)} = 1 + D + D^2 + D^3$, which can be written in octal form: $(13, 17)$. In Fig. 1 the path shown in bold corresponds to the beginning of one of the codewords. For a detailed explanation of the trellis concept, see [18].

2.2. Active Distances and Spectrum of Active Distances

One of the most important distance properties of a convolutional code is its free distance.

Definition 1. The free distance of a convolutional code is the smallest weight of a nonzero codeword.

Except for the free distance, convolutional codes have other important distance characteristics that determine the probability of erroneous decoding.

For block codes, the weight spectrum of a code is usually used to estimate the FER [24]. In this paper, we suggest to use active distances proposed in [25] and the spectrum of active distances for the convolutional code analysis. If a usual weight spectrum of a convolutional code is considered, then weights are nonincreasing while the codeword length grows. This situation occurs if the codeword path merges the zero path after some time moment. That is why it is reasonable to exclude such codewords from consideration after they merge the zero path. This is the main difference between these two definitions: weight spectrum and spectrum of active distances.

To start with, we define an additional construction to further consider active distances.

Definition 2. The subset of codewords of a convolutional code $\mathcal{C}$ with one finite nonzero branch is denoted by $\mathcal{C}_f$. This subset $\mathcal{C}_f$ consists of codewords $\boldsymbol{v}(D)$ which have a corresponding finite nonzero branch of some length j on the trellis path. This nonzero branch begins right after the initial zero state: $s_0 = 0$, $s_1 \neq 0$, and does not have two consecutive zero trellis states before merging with the zero path $s_j = s_{j+1} = s_{j+2} = \ldots = 0$ at some time moment j.

Codewords from the subset $\mathcal{C}_f$ have finite weight, and this weight for each codeword equals the weight of the nonzero branch. When defining active distances, we consider only this first nonzero part of a codeword. The subset $\mathcal{C}_f$ does not contain the zero codeword. The minimum length of a nonzero path is $m+1$, where m is the memory of the convolutional code. Hereinafter, the trellis length is considered in terms of the number of output tuples (or the number of transitions between trellis states). Then we denote a codeword from the subset $\mathcal{C}_f$ with nonzero trellis path of length j by $\boldsymbol{v}^{(j)}$.

Definition 3. The active distance of length j for a convolutional code $\mathcal{C}$ is the minimum Hamming weight of a codeword $\boldsymbol{v}^{(j)}$ from the set $\mathcal{C}_f$ that has a nonzero trellis path of length j:

$$a_j = \min_{\boldsymbol{v}^{(j)} \in \mathcal{C}_f} w(\boldsymbol{v}^{(j)}),$$

where $w(\boldsymbol{v}^{(j)})$ is the Hamming weight of $\boldsymbol{v}^{(j)}$.

The free distance of a convolutional code can also be defined in terms of active distances, since when considering the subset $\mathcal{C}_f$ we do not exclude codewords with the smallest nonzero weight. Then the free distance can be introduced as the minimal active distance $\min\limits_j a_j$, $j \geq m+1$. It is important to note that the free distance need not always correspond to the smallest length j.

Since the weight of a codeword $\boldsymbol{v}^{(j)} \in \mathcal{C}_f$ is determined only by the weight of the nonzero branch, the weight of the codeword can be given using the first j output tuples:

$$w(\boldsymbol{v}^{(j)}) = \sum_{i=0}^{j-1} w(\boldsymbol{v}_i^{(j)}).$$

We denote the number of codewords $\boldsymbol{v}^{(j)} \in \mathcal{C}_f$ with a fixed weight $w^{(j)}$ by $N_{w^{(j)}}$. Let us present the definition of the spectrum of active distances, which was first proposed in [21].

Definition 4. The spectrum $\mathcal{D}_{a_j}$ of active distances a_j of length j for a convolutional code is the set

$$\mathcal{D}_{a_j} = \left\{ j, a_j, w^{(j)}, N_{w^{(j)}} \mid w^{(j)} = w(\boldsymbol{v}^{(j)}),\ \boldsymbol{v}^{(j)} \in \mathcal{C}_f \right\}.$$

The minimum weight in the spectrum of active distances $\mathcal{D}_{a_j}$ is the active distance a_j of length j of the convolutional code according to the definition of the active distance. The calculation of the code spectrum is more computationally intensive than finding the spectrum of active distances. This is due to the exclusion of paths that follow the zero path when computing the spectrum of active distances. The union of spectra of active distances of length j over all possible lengths forms the whole spectrum of active distances.

Definition 5. The spectrum of active distances $\mathcal{D}_a$ for a convolutional code is the union of the sets $\mathcal{D}_{a_j}$ over all possible lengths j:

$$\mathcal{D}_a = \bigcup_{j=m+1}^{\infty} \mathcal{D}_{a_j},$$

where m is the code memory.

The spectrum of active distances can be represented as a table. Here we provide an example of spectra of active distances for two convolutional codes.

Example 1. In Tables 1 and 2 we present the spectrum of active distances for two convolutional codes with a recursive encoder and memory $m = 3$: $(13, 15)$ and $(13, 17)$. The number in the table

Table 1. Spectrum of active distances for the $(13, 15)$ convolutional code with recursive encoder.

$w\backslash j$	4	5	6	7	8	9	10	11	12	13	14	15
6	**1**		**1**									
7												
8		**1**	1	**2**	**2**	**2**	**1**	**1**				
9												
10				2	4	6	8	8	**8**	**6**	**4**	**2**
11												
12					2	5	13	19	29	34	36	34
13												
14						2	6	20	38	68	100	132
15												
16							1	8	25	64	132	230
17												
18									8	30	93	220
19												
20										6	32	121
21												
22											4	32
23												
24												2
25												

Table 2. Spectrum of active distances for the $(13, 17)$ convolutional code with recursive encoder.

$w\backslash j$	4	5	6	7	8	9	10	11	12	13	14	15
6		**1**										
7	**1**		**1**	**1**								
8			1	1	**1**	**2**						
9					4	1	**3**	**3**				
10				1	1	3	6	4	**5**	**5**		
11				1		4	5	5	15	7	**10**	**8**
12					1	2	4	13	10	18	27	16
13						3	3	13	14	30	32	37
14					1		5	7	22	34	40	81
15							3	5	15	40	60	96
16								2	17	21	79	105
17								2	7	20	62	119
18								2		21	36	100
19									2	7	26	90
20										5	12	71
21									1		11	27
22											6	11
23												6
24												3
25												3

at the intersection of the ith row and jth column denotes the number of codewords in the subset $\mathcal{C}_f$ with the weight shown in the ith row and the nonzero branch length shown in the jth column. The spectrum coefficients with weights corresponding to active distances are set off in bold.

3. THEORETICAL ESTIMATES OF THE FRAME AND BIT ERROR RATES FOR CONVOLUTIONAL CODES IN A BINARY SYMMETRIC CHANNEL

Here we provide upper and lower estimates for the probability of error burst and the frame error rate and also derive an estimate for the bit error rate. It is assumed that the data transmission is performed over a binary symmetric channel (BSC) and that the Viterbi algorithm is used for the decoding.

3.1. Probability of Error Burst

The probability of error burst and its estimate were described in detail in [21]. Here we provide the derivation and the final formula.

We consider a BSC with maximum likelihood decoding and crossover probability p. The Viterbi decoder always returns a codeword that belongs to the original convolutional code $\mathcal{C}$. If an error burst occurs while decoding, then the decoded codeword $\boldsymbol{v}' \in \mathcal{C}$ differs from the transmitted codeword $\boldsymbol{v} \in \mathcal{C}$. At the place of the error burst there is an offset from the correct trellis path. Two error bursts in one codeword are independent if between the corresponding branches there are at least two correct consecutive states on the trellis. Since the convolutional code is linear, we have $\boldsymbol{v} + \boldsymbol{v}' \in \mathcal{C}$, and at the place of the error burst in this sum there is a nonzero branch with weight defined by the spectrum $\mathcal{D}_a$ of active distances of the convolutional code. In the decoded codeword, an error burst occurs if there are at least $\frac{a_j}{2}$ errors at the position of ones in the sum $\boldsymbol{v} + \boldsymbol{v}'$ at the place of the error burst, where a_j is the minimum weight of the error burst of length j (the active distance).

The probability of a fixed number of errors at the positions of ones and zeros in a binary sequence can be given as follows:

$$P(e_{\text{all}}, e_1, j, w, p) = \binom{w}{e_1}\binom{2j-w}{e_{\text{all}}-e_1} p^{e_{\text{all}}}(1-p)^{2j-e_{\text{all}}},$$

where e_{all} is the total number of errors, e_1 the number of errors between ones, j the length of the sequence in terms of the number of tuples, w the weight of the sequence, and p the crossover probability in the channel.

As was mentioned above, an error burst occurs if the number of errors between ones in the burst is greater than half the weight, and the number of errors between zeros can be arbitrary. If the number of errors between ones in the burst is exactly equal to half the weight, the error burst occurs with probability 1/2. Thus, the probability of an error burst of a given length j and weight w for the binary symmetric channel with crossover probability p can be written as follows:

$$\begin{aligned} P_{\text{burst}}(j, w, p) = &\sum_{i>\frac{w}{2}}^{2j} \sum_{i_1>\frac{w}{2}}^{\min(w,i)} P(e_{\text{all}} = i, e_1 = i_1, j, w, p) \\ &+ \begin{cases} 0, & \text{for odd } w, \\ \dfrac{1}{2} \displaystyle\sum_{i=\frac{w}{2}}^{2j-\frac{w}{2}} P\Big(e_{\text{all}} = i, e_1 = \dfrac{w}{2}, j, w, p\Big), & \text{for even } w. \end{cases} \end{aligned} \tag{2}$$

We compute a lower estimate for the probability of error burst as the probability of the most likely error burst, i.e., the burst with the smallest weight. Then an upper estimate can be found as the sum of the probabilities of error bursts with all possible lengths. The upper and lower estimates for the probability of an error burst of length j can be given using active distances and the spectrum of active distances, respectively:

$$P_{\text{low}}(j, p) = (1-p)^{4m} P_{\text{burst}}(j, w = a_j, p) \tag{3}$$

and

$$P_{\text{up}}(j, p) = \sum_{w_i = w_{\min}}^{w_{\max}} N_{w_i} P_{\text{burst}}(j, w = w_i, p), \tag{4}$$

where $\{w_i, N_{w_i}\} \in \mathcal{D}_{a_j}$ and m is the code memory. For the lower estimate, we also use the factor $(1-p)^{4m}$, i.e., the probability of $2m$ correct bits before and after the burst, to exclude situations where the considered sequence is a part of some larger burst. A more detailed explanation can be found in [23].

3.2. Probability of Erroneous Decoding

A codeword of a convolutional code has infinite length in the general case. In real-life tasks, the data size is finite and terminated convolutional codes are used. We use zero-tailed (zero terminated) convolutional codes, but there are also tail-biting convolutional coding as a possible solution. Blocks of finite length are transmitted through the channel. They consist of a sequence of output tuples of the terminated convolutional code, each of them corresponding to a finite trellis path with zero initial and final states. Since transition from any state to the zero state may take at most m time steps (while successively writing zeros to the register), where m is the code memory, code termination slightly reduces the code rate. Thus, we transmit m information bits less, which is usually small as compared to the block length $L \gg m$.

We estimate the probability of erroneous decoding of a block of length L using the average fraction of blocks with at least one error burst. A lower estimate for the erroneous decoding probability, i.e., for the frame error rate (FER), will be obtained as the average fraction of blocks with the most likely error burst. This burst has the minimum possible weight $w_{\min} = \min_j a_j$. For the probability of error burst we use the lower estimate (3) for $P_{\text{low}}(j,p)$. To obtain a more accurate lower estimate, we multiply the probability of the most likely error burst by the number of such different bursts $N_{w_{\min}}$; multiplying this by the block length L in terms of the number of tuples, we obtain a lower estimate for the FER:

$$\text{FER}_{\text{low}}(p) = LN_{w_{\min}} P_{\text{low}}(j = j_{w_{\min}}, p), \tag{5}$$

where $j_{w_{\min}}$ is the length of the burst with the minimum weight, $\{w_{\min}, N_{w_{\min}}\} \in \mathcal{D}_a$.

An upper estimate is derived as an additive bound by summing up the average fractions of blocks affected by bursts of different lengths. As the error burst probability, we use the upper estimate (4) for $P_{\text{up}}(j,p)$. Since the number of bursts is already involved in this estimate, there is no need in the additional multiplier. Thus, we obtain an upper estimate for the FER:

$$\text{FER}_{\text{up}}(p) = \min\left\{1, \sum_{\ell=m+1}^{L} LP_{\text{up}}(j = \ell, p)\right\}, \tag{6}$$

where m is the code memory.

The upper estimate can be slightly improved, but since the probability of a block with two or more bursts is in several orders of magnitude smaller (since the probabilities of bursts are multiplied separately), we for simplicity take such blocks into account several times in our additive estimate. The sum in (6) is over all possible bursts lengths, but for practical use it suffices to take only the shortest bursts into account, which can be seen from Fig. 2. Even for few shortest bursts, the estimate $\text{FER}_{\text{up}}(p)$ lies above the FER simulation curve. For small p, considering a larger number of different burst lengths has almost no effect on the estimation accuracy.

3.3. Probability of Error per Bit

The output probability of error per bit for a convolutional code is the fraction of erroneous bits in the decoded codeword. The erroneous bits may occur in positions of error bursts only.

We estimate the probability of error per bit, i.e., the bit error rate (BER), as the average fraction of erroneous bits. We derive a lower estimate for the BER as the fraction of erroneous bits assuming that the most likely burst occurs. Then the lower estimate is the weight of the most likely burst multiplied by the probability of its occurrence in a fixed bit. For a more accurate lower bound, as the probability of error burst with the smallest weight $w_{\min} = \min_j a_j$ we use the lower estimate for the burst probability in a tuple (3) multiplied by the number of such bursts (they are equiprobable) and normalized by the number of bits in a tuple. Thus, the proposed lower estimate for the BER is

$$\text{BER}_{\text{low}}(p) = \frac{w_{\min} N_{w_{\min}}}{2} P_{\text{low}}(j = j_{w_{\min}}, p), \tag{7}$$

where $N_{w_{\min}}$ is the number of bursts with the minimum weight, $j_{w_{\min}}$ is the length of a burst with the minimum weight, $\{w_{\min}, N_{w_{\min}}\} \in \mathcal{D}_a$, and the denominator $n = 2$ is the number of bits in a tuple.

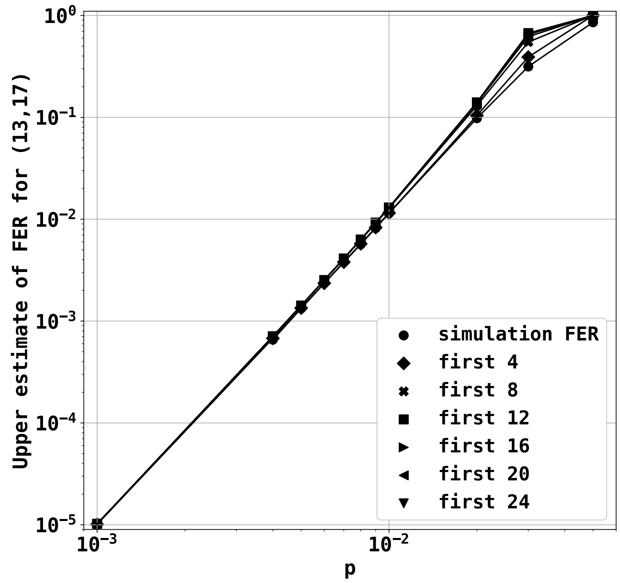

Fig. 2. Burst length j versus the FER upper estimate for the $(13, 17)$ convolutional code with recursive systematic encoder and memory $m = 3$.

An upper estimate for the BER is constructed as an additive bound over the average fractions of erroneous bits for all lengths of bursts with the maximum weight. The upper estimate for the error burst probability in a tuple (4) takes into account the number of such bursts, so there is no need in an additional multiplier. As in the case of the estimate $\mathrm{BER}_{\mathrm{low}}$, we normalize the burst probability by the number $n = 2$ of bits in a tuple to obtain the probability of occurrence of a burst in a fixed bit. Then the BER upper estimate is

$$\mathrm{BER}_{\mathrm{up}}(p) = \min\left\{1, \sum_j \frac{\max_{w_j \in \mathcal{D}_{a_j}} w_j}{2} P_{\mathrm{up}}(j, p)\right\}. \tag{8}$$

In expression (8) we take the minimum of 1 and the estimate, since the estimate is additive and may exceed 1. In (8) the sum is over all possible values of j, while for practical use it suffices to take the sum over the shortest bursts only. Indeed, it can be seen from Fig. 3 that the upper estimate converges as the summation index j in the sum (8) grows. Moreover, even for a small number of considered burst lengths, the estimate lies above the simulation curve. For small per bit crossover probabilities p ($p \leq 0.01$), taking few shortest bursts is sufficient for a rather accurate estimation, which also reduces the computational complexity.

3.4. Complexity of Theoretical Estimates

For the computation of the proposed estimates, one has to know active distances and the spectrum of active distances for a convolutional code. Note that the distance properties of the convolutional code should be calculated only once.

Computation of the active distances a_j for length j is based on the Viterbi algorithm and proceeds as follows:

1. At the initial time moment $t = 0$, initialize one path at the zero trellis state of length 0.
2. At time moment $t > 0$, continue all saved paths over the trellis from states at time moment $t-1$ to all possible states at time moment t. Each path can be continued in 2^k ways, $k = 1$, without taking into account paths from the zero trellis state. From the zero trellis state, the path can be

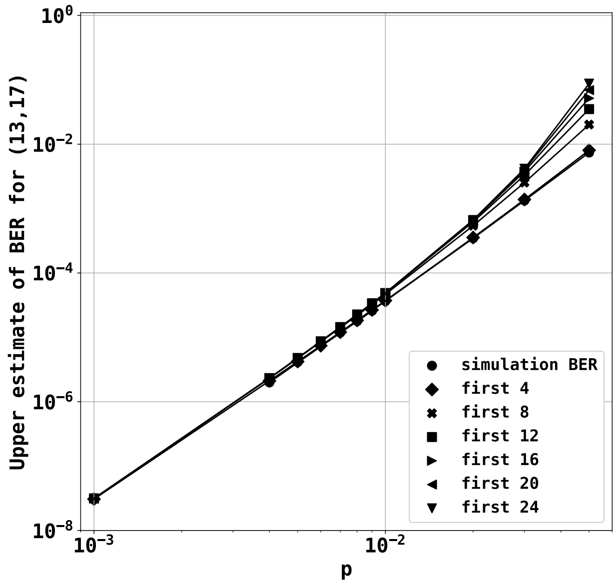

Fig. 3. Burst length j versus the BER upper estimate for the $(13, 17)$ convolutional code with recursive systematic encoder and memory $m = 3$.

continued only in the next nonzero state. Calculate the weights of the paths, the total number of paths being $2^{m+k} - 1$, $k = 1$ (if $t > m$, where m is the code memory).

3. For each trellis state at time moment t, save only the path with the minimum weight; 2^m paths remain (if $t \geq m$).
4. If $t = j$, return the weight of the path saved in the zero state at time moment $t = j$. If $t < j$, go to Step 2.

It is convenient to represent the computation process for the active distances as moving to the right along the trellis. Since the task is to find a nonzero trellis path of length j with the minimum weight, it suffices to save for each state at each time moment only the path with the smallest weight that came to this state. The complexity of this algorithm depends on the length j and on the number of trellis states 2^m, where m is the code memory. It follows from the algorithm that the complexity of computing the active distances a_j is $\mathcal{O}(j(2^{m+1} - 1))$. Also, while computing the active distances of length j, we can obtain all active distances for lengths less than j.

The computation of the spectrum of active distances has larger computational complexity, which grows exponentially with the length j of the nonzero path for which the spectrum has to be computed. The spectrum of active distances is also computed over the trellis with the only difference that we save all paths rather than choose one for the state with the minimal weight. The path from the zero state can only be continued to a nonzero state. The computational complexity of the algorithm can be written as $\mathcal{O}(2^j(2^{m+1} - 1))$, where m is the code memory. From this expression it is clear that computing the spectrum for large lengths is computationally intensive and grows rapidly with j.

Now we estimate the complexity of the proposed estimates for the error probability for convolutional codes if distance properties (active distances and the spectrum of active distances) are already known. The complexity of computation of the lower, $P_{\text{low}}(j,p)$, (3), and upper, $P_{\text{up}}(j,p)$, (4), estimates for the error burst probability for length j is found as the maximum number of computations of $P(e_{\text{all}}, e_1, j, w, p)$:

$$\mathcal{O}\Big(\Big(2j - \frac{a_j}{2}\Big)\frac{a_j}{2}\Big) \qquad \text{and} \qquad \mathcal{O}\Big(\Big(2j - \frac{w_{\min}}{2}\Big)\frac{w_{\max}}{2}(w_{\max} - w_{\min})\Big), \quad \{w_{\min}, w_{\max}\} \in \mathcal{D}_{a_j}.$$

The complexity of computing lower estimates (5) and (7) for $\mathrm{FER}_{\mathrm{low}}(p)$ and $\mathrm{BER}_{\mathrm{low}}(p)$ does not differ from the complexity of $P_{\mathrm{low}}(j = j_{w_{\min}} p)$, where $j_{w_{\min}}$ is the length of a burst with the minimal weight. The complexity of upper estimates (6) and (8) for $\mathrm{FER}_{\mathrm{up}}(p)$ and $\mathrm{BER}_{\mathrm{up}}(p)$ coincides with the complexity of computing the upper estimate for the probability of error burst $P_{\mathrm{up}}(j, p)$ multiplied by the number of considered possible weights of error bursts. In practice, it suffices to consider about 5–8 different weights, as follows from Fig. 3. Thus, although the computational complexity of theoretical estimates does not depend on the crossover probability p, for practical values of p it suffices to consider only a few number of terms in the sum for rather accurate error probability values. Also, the complexity of the theoretical estimates is linear in the length j.

3.5. Experimental Results

Here we present simulation results and their comparison to the theoretical estimates. In simulations we consider the BSC with crossover probability p. Random codewords of length 1000 tuples (2000 bits) of a terminated convolutional code with recursive systematic encoder and rate 1/2 were transmitted over the channel. We considered different convolutional codes with different code memories.

First, we provide results for the probability of error burst. The curves in Fig. 4 are presented for three different crossover probabilities p: 0.03, 0.008, and 0.004. For $p = 0.004$, we provide less simulation points, since the time needed for obtaining the results grows rapidly for small crossover probabilities.

It follows from Fig. 4 that the probability estimates for short error bursts are the nearest to the simulation results, since for these lengths there are only one or several codewords into which a codeword can be changed. When the burst length increases, the number of possible codewords to be obtained while decoding grows rapidly. This can also be seen from Tables 1 and 2 for the spectrum of active distances. Bursts of the shortest lengths have the highest probability and the greatest impact in the FER and BER. From Fig. 4 we can see that while the crossover probability p decreases, the estimates converge, which can be explained by the exponential decrease in the probability of long error bursts. Thus, for small crossover probabilities p, the proposed estimates are rather accurate even for long bursts.

It is known that the distribution of error bursts can be approximated with the geometric distribution [26, 27]. This holds only for the least likely bursts; i.e., the geometric distribution is a good estimate for the probability of long bursts and is bad for the most likely bursts of small lengths.

In Figs. 5 and 6, FER and BER results for two convolutional codes, $(13, 17)$ and $(13, 15)$, with memory $m = 3$ are presented. We also provide BER results for the $(117, 155)$ code with memory $m = 6$ in Fig. 7 to prove that our estimates are also applicable to codes with larger memories. For comparison, we also present the classic additive upper bounds on the FER and BER from [14, 15] computed using the generating function for the code spectrum and the function for the per bit error, respectively.

Estimates for the probability of erroneous decoding were also presented in [23]. These results can be compared to those from [20] for the $(13, 17)$ code, and it is clear that they are almost the same. Nevertheless, the method from [20] is computationally intensive, especially for large code memories. Moreover, the authors of [20] mention that their approach is inapplicable to codes with memory greater than 4.

The method proposed in the present article, based on active distances of a convolutional code and their spectrum, allows to compute estimates for codes with larger memory without using additional computing resources. To obtain tight estimates, it is not required to consider long bursts; considering several shortest bursts suffices. Computer programs for the computation of

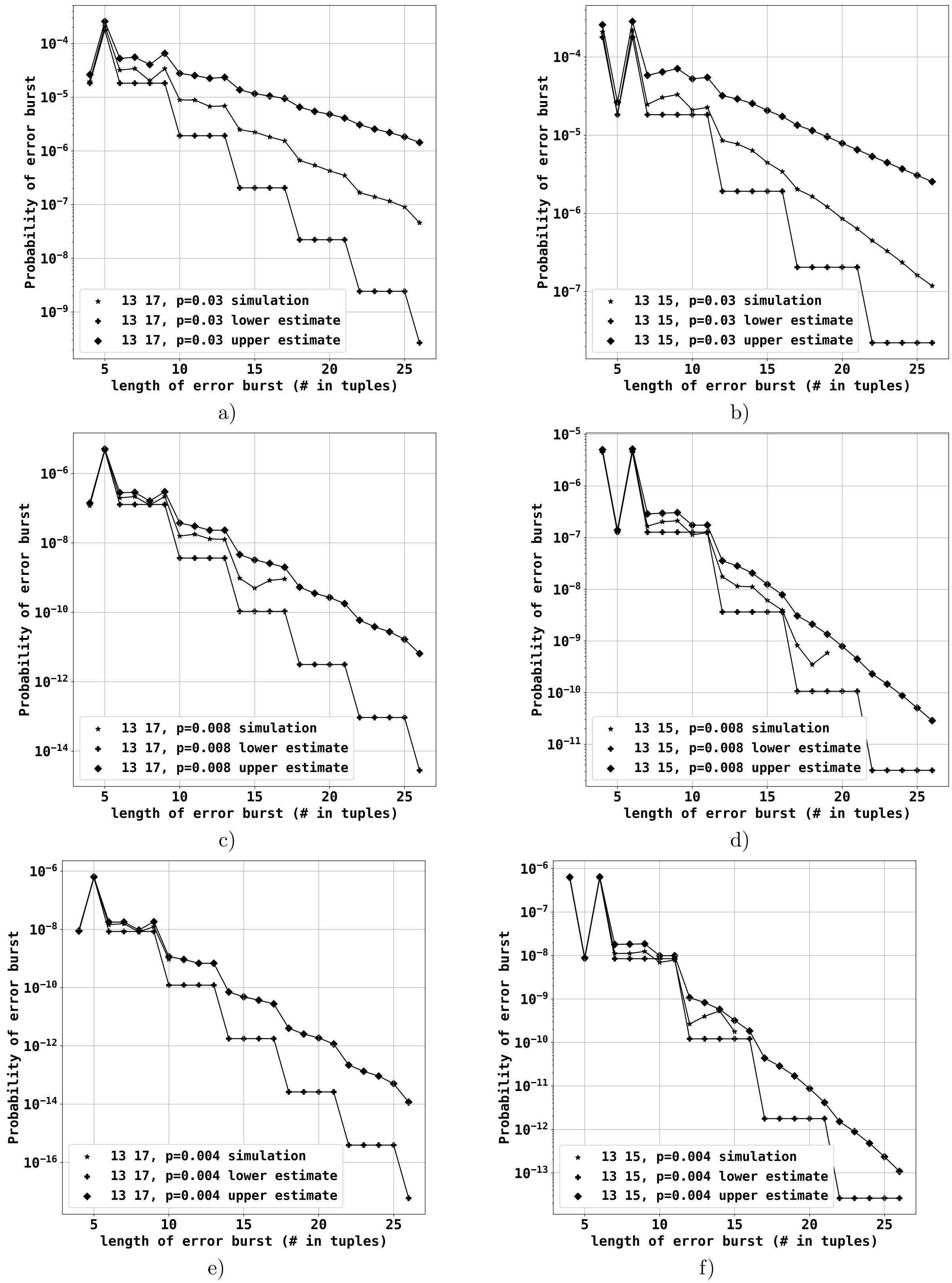

Fig. 4. Probability distribution of error bursts for convolutional codes with recursive systematic encoder and memory $m = 3$: a) $(13, 17)$, $p = 0.03$, b) $(13, 15)$, $p = 0.03$, c) $(13, 17)$, $p = 0.008$, d) $(13, 15)$, $p = 0.008$, e) $(13, 17)$, $p = 0.004$, f) $(13, 15)$, $p = 0.004$.

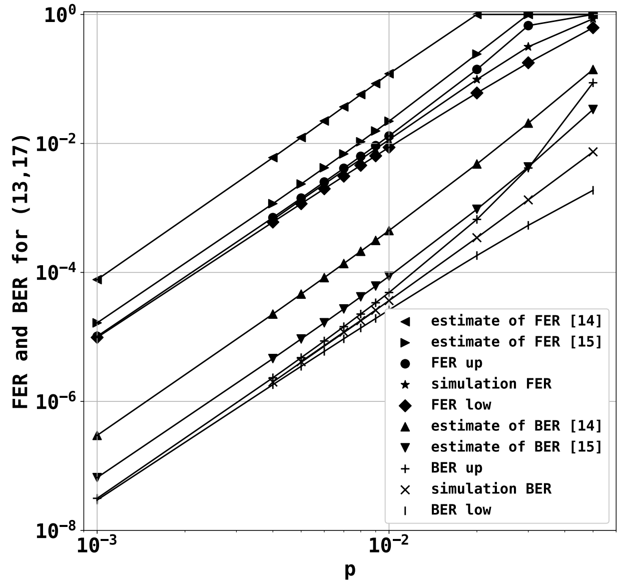

Fig. 5. FER and BER for the $(13, 17)$ convolutional code with recursive systematic encoder and memory $m = 3$.

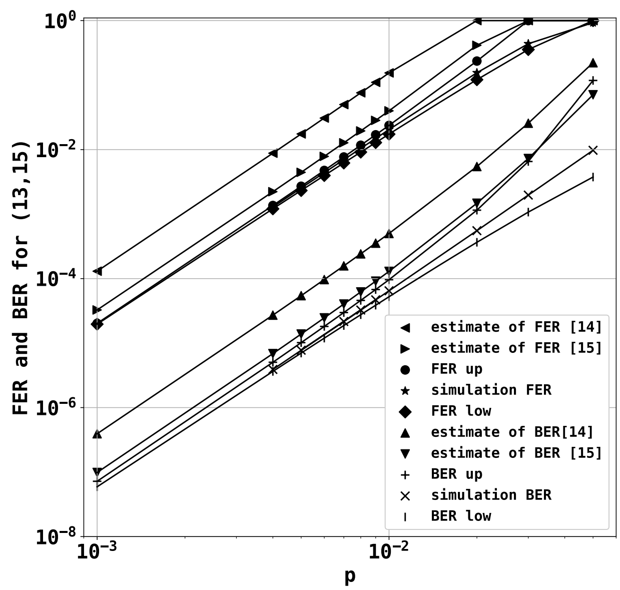

Fig. 6. FER and BER for the $(13, 15)$ convolutional code with recursive systematic encoder and memory $m = 3$.

active distances and their spectrum are provided on GitHub [28]. Here we also present an example of the BER for the $(117, 155)$ convolutional code with memory $m = 6$ in Fig. 7, where BER upper estimates from [14, 15] are also presented for comparison.

From Figs. 5–7 it is clear that the upper and lower estimates are close to each other for small crossover probabilities p. Thus, for these probabilities theoretical estimates are rather accurate and there is no need to use the Monte-Carlo method to obtain experimental values, which in practice might require much resources for small crossover probabilities p. We also compare our upper estimates for the FER and BER with classical upper estimates from [14, 15] obtained with the use of generating functions.

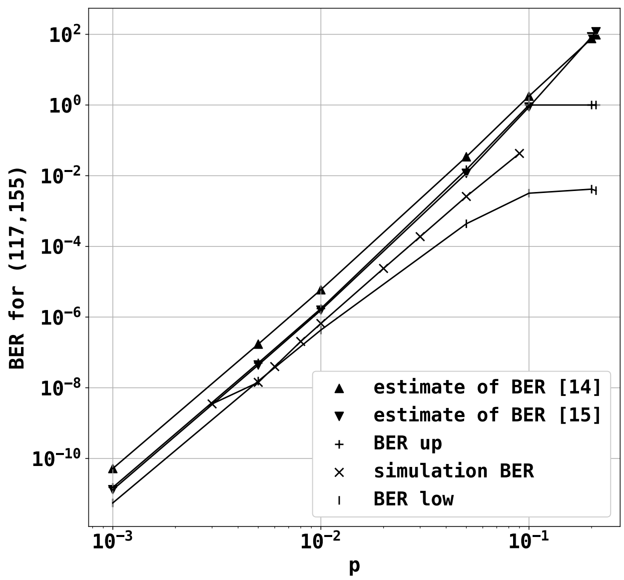

Fig. 7. BER for the $(117, 155)$ convolutional code with recursive systematic encoder and memory $m = 6$.

3.6. Comparison with Other Bounds

Estimates for the probability of erroneous decoding were also presented in [23]. These results can be compared to those from [20] for the $(13, 17)$ code, and it is clear that they are almost the same. Nevertheless, the method from [20] is computationally intensive, especially for large code memories. Moreover, the authors of [20] mention that their approach is inapplicable to codes with memory greater than 4, while our estimates are shown to be applicable to codes with larger memory too.

The method proposed in the present article, based on active distances of a convolutional code and their spectrum, allows to compute estimates for codes with larger memory without using additional computing resources. To obtain tight estimates, it is not required to consider long bursts; considering several shortest bursts suffices. Computer programs for the computation of active distances and their spectrum can be found on GitHub [28].

In Figs. 5–7 we present a comparison of the proposed estimates with the classical bounds from [14, 15]. From the results it follows that the additive upper estimate proposed here is closer than or comparable with the classical bounds. The estimate from [15] is closer to our bound and to the simulation results than the estimate from [14], which can be explained by a more accurate estimate using Meeberg's bound for the probability of error in the BSC. The estimates for the probability of error burst and active distances that we use allow to compute estimates for the FER and BER more accurately. We also note that we also derive here a lower estimate, which was not proposed earlier in the literature, and the lower estimate lies very near to the simulation results.

4. CONCLUSION

In this paper we have obtained most important theoretical estimates for the performance of convolutional codes in the binary symmetric channel. Namely, we derived upper and lower estimates for the error burst probability, FER, and BER. The proposed estimates are based on active distances and the spectrum of active distances, and they are valid for convolutional codes of rate $1/2$ with Viterbi decoding. The obtained results can be generalized to convolutional codes of rate $1/n$. We analyzed the computational complexity for distance properties and theoretical estimates. It was shown that for a given length the computational complexity for active distances is exponential in the code memory and linear in the length, while the complexity of computing the spectrum of active distances is exponential in both the code memory and length. The time complexity for the theoretical estimates with known distance properties is linear in the length of the shortest error bursts and does not depend on the crossover probability in the channel. Also, we considered various codes with various memories and presented a comparison of theoretical and simulation results for these codes. The proposed estimates are the most tight for small crossover probabilities.

In further work, estimates for the performance of convolutional codes in a Gaussian noise channel can also be derived using active distances and their spectra.

FUNDING

The research was funded under the HSE University Basic Research Program in 2022.

REFERENCES

1. Elias, P., Coding for Noisy Channels, *IRE Conv. Rec.*, 1955, vol. 4, pp. 37–46. Reprinted in *Key Papers in the Development of Information Theory*, Slepian, D., Ed., New York: IEEE Press, 1974, pp. 102–111.

2. Yang, C., Zhan, M., Deng, Y., Wang, M., Luo, X.H., and Zeng, J., Error-Correcting Performance Comparison for Polar Codes, LDPC Codes and Convolutional Codes in High-Performance Wireless, in *Proc. 6th Int. Conf. on Information, Cybernetics, and Computational Social Systems (ICCSS'2019), Chongqing, China, Sept. 27–30, 2019*, P. 258–262. https://doi.org/10.1109/ICCSS48103.2019.9115442

3. Deng, Y., Zhan, M., Wang, M., Yang, C., Luo, X., Zeng, J., and Guo, J., Comparing Decoding Performance of LDPC Codes and Convolutional Codes for Short Packet Transmission, in *Proc. IEEE 17th Int. Conf. on Industrial Informatics (INDIN'2019), Helsinki, Finland, July 22–25, 2019*, vol. 1, pp. 1751–1755. https://doi.org/10.1109/INDIN41052.2019.8972062

4. Tahir, B., Schwarz, S., and Rupp, M., BER Comparison between Convolutional, Turbo, LDPC, and Polar Codes, in *Proc. 24th Int. Conf. on Telecommunications (ICT'2017), Limassol, Cyprus, May 3–5, 2017*, pp. 1–7. https://doi.org/10.1109/ICT.2017.7998249

5. Rurik, W. and Mazumdar, A., Hamming Codes as Error-Reducing Codes, in *Proc. 2016 IEEE Information Theory Workshop (ITW'2006), Cambridge, UK, Sept. 11–14, 2016*, pp. 404–408. https://doi.org/10.1109/ITW.2016.7606865

6. Liu, K. and García-Frías, J., Error Floor Analysis in LDGM Codes, in *Proc. 2010 IEEE Int. Symp. on Information Theory (ISIT'2010), Austin, TX, USA, June 13–18, 2010*, pp. 734–738. https://doi.org/10.1109/ISIT.2010.5513607

7. Gao, W. and Polyanskiy, Y., On the Bit Error Rate of Repeated Error-Correcting Codes, in *Proc. 48th Annu. Conf. on Information Sciences and Systems (CISS'2014), Princeton, NJ, USA, Mar. 19–21, 2014*, pp. 1–6. https://doi.org/10.1109/CISS.2014.6814087

8. Höst, S., Johannesson, R., and Zyablov, V.V., Woven Convolutional Codes I: Encoder Properties, *IEEE Trans. Inform. Theory*, 2002, vol. 48, no. 1, pp. 149–161. https://doi.org/10.1109/18.971745

9. Freudenberger, J., Bossert, M., Shavgulidze, S., and Zyablov, V., Woven Turbo Codes, in *Proc. 7th Int. Workshop on Algebraic and Combinatorial Coding Theory (ACCT'2000), Bansko, Bulgaria, June 18–24, 2000*, pp. 145–150. Available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.23.345&rep=rep1&type=pdf

10. Benedetto, S. and Montorsi, G., Design of Parallel Concatenated Convolutional Codes, *IEEE Trans. Commun.*, 1996, vol. 44, no. 5, pp. 591–600. https://doi.org/10.1109/26.494303

11. Berrou, C., Glavieux, A., and Thitimajshima, P., Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes. 1, in *Proc. IEEE Int. Conf. on Communications (ICC'93), Geneva, Switzerland, May 23–26, 1993*, vol. 2, pp. 1064–1070. https://doi.org/10.1109/ICC.1993.397441

12. Douillard, C. and Berrou, C., Turbo Codes with Rate-$m/(m+1)$ Constituent Convolutional Codes, *IEEE Trans. Commun.*, 2005, vol. 53, no. 10, pp. 1630–1638. https://doi.org/10.1109/TCOMM.2005.857165

13. Zhang, Q., Liu, A., Zhang, Y., and Liang, X., Practical Design and Decoding of Parallel Concatenated Structure for Systematic Polar Codes, *IEEE Trans. Commun.*, 2016, vol. 64, no. 2, pp. 456–466. https://doi.org/10.1109/TCOMM.2015.2502246

14. Viterbi, A.J., Convolutional Codes and Their Performance in Communication Systems, *IEEE Trans. Commun. Technol.*, 1971, vol. 19, no. 5, pp. 751–772. https://doi.org/10.1109/TCOM.1971.1090700

15. van De Meeberg, L., A Tightened Upper Bound on the Error Probability of Binary Convolutional Codes with Viterbi Decoding, *IEEE Trans. Inform. Theory*, 1974, vol. 20, no. 3, pp. 389–391. https://doi.org/10.1109/TIT.1974.1055216

16. Herro, M., Hu, L., and Nowack, J., Bit Error Probability Calculations for Convolutional Codes with Short Constraint Lengths on Very Noisy Channels, *IEEE Trans. Commun.*, 1988, vol. 36, no. 7, pp. 885–888. https://doi.org/10.1109/26.2819

17. Chiaraluce, F., Gambi, E., Mazzone, M., and Pierleoni, P., A Technique to Evaluate an Exact Formula for the Bit Error Rate of Convolutional Codes in Case of Finite Length Words, *Proc. IEEE Region 10 Annu. Conf. on Speech and Image Technologies for Computing and Telecommunications (IEEE TENCON'97), Queensland Univ. of Technology, Brisbane, Australia, Dec. 2–4, 1997*, Deriche, M., Moody, M., and Bennamoun, M., Eds., vol. 1, pp. 113–116. https://doi.org/10.1109/TENCON.1997.647271

18. Forney, G., The Viterbi Algorithm, *Proc. IEEE*, 1973, vol. 61, no. 3, pp. 268–278. https://doi.org/10.1109/PROC.1973.9030

19. Yoshikawa, H., Theoretical Analysis of Bit Error Probability for Punctured Convolutional Codes, in *Proc. 2012 IEEE Int. Sympos. on Information Theory and Its Applications (ISITA'2012), Honolulu, HI, USA, Oct. 28–31, 2012*, pp. 658–661.

20. Bocharova, I.E., Hug, F., Johannesson, R., and Kudryashov, B.D., A Closed-Form Expression for the Exact Bit Error Probability for Viterbi Decoding of Convolutional Codes, *IEEE Trans. Inform. Theory*, 2012, vol. 58, no. 7, pp. 4635–4644. https://doi.org/10.1109/TIT.2012.2193375

21. Smeshko, A., Ivanov, F., and Zyablov, V., Theoretical Estimates of Burst Error Probability for Convolutional Codes, in *Proc. 2020 Int. Symp. on Information Theory and Its Applications (ISITA'2020), Kapolei, HI, USA, Oct. 24–27, 2020*, pp. 136–140.

22. Smeshko, A., Ivanov, F., and Zyablov, V., The Influence of Active Distances on the Distribution of Bursts, in *Proc. XVI Int. Symp. "Problems of Redundancy in Information and Control Systems" (REDUNDANCY'2019), Moscow, Russia, Oct. 21–25, 2019*, pp. 110–114. https://doi.org/10.1109/REDUNDANCY48165.2019.9003349

23. Smeshko, A., Ivanov, F., and Zyablov, V., Upper and Lower Estimates of Frame Error Rate for Convolutional Codes, *Proc. 2020 Int. Symp. on Information Theory and Its Applications (ISITA'2020), Kapolei, HI, USA, Oct. 24–27, 2020*, pp. 160–164.

24. Kudryashov, B.D., *Osnovy teorii kodirovaniya* (Fundamentals of Coding Theory), St. Petersburg: BHV-Petersburg, 2016.

25. Höst, S., Johannesson, R., Zigangirov, K., and Zyablov, V., Active Distances for Convolutional Codes, *IEEE Trans. Inform. Theory*, 1999, vol. 45, no. 2, pp. 658–669. https://doi.org/10.1109/18.749009

26. Miller, R.L., Deutsch, L.J., and Butman, S.A., On the Error Statistics of Viterbi Decoding and the Performance of Concatenated Codes, *NASA STI/Recon Tech. Rep.*, Sept. 1, 1981, no. 81-33364. Available at https://archive.org/details/nasa_techdoc_19810024821

27. Justesen, J. and Andersen, J., Critical Lengths of Error Events in Convolutional Codes, *IEEE Trans. Inform. Theory*, 1998, vol. 44, no. 4, pp. 1608–1611. https://doi.org/10.1109/18.681339

28. https://github.com/smeshk/Active-distances-their-spectrum-and-estimates-for-convolutional-code [GitHub online repository], 2021.