NON-STATISTICAL EUCLIDEAN-DISTANCE SISO DECODING OF ERROR-CORRECTING CODES OVER GAUSSIAN AND OTHER CHANNELS

Mónica C. Liberatori[1], Leonardo J. Arnone[1], Jorge Castiñeira Moreira[1] and Patrick G. Farrell[2]

[1]Electronics Engineering Department, ICYTE, Mar del Plata University and CONICET
[2] Kent University, UK

ABSTRACT

In this paper we describe novel non-statistical Euclidean distance soft-input, soft-output (SISO) decoding algorithms for the three currently most important error-correcting codes: the low-density parity-check (LDPC), turbo and polar codes. The metric is squared Euclidean distance, and the decoders operate using an antilog-log (AL) process. We have investigated the simulated bit-error rate (BER) performance of these non-statistical algorithms on three channel models: the additive White Gaussian noise (AWGN), the Rayleigh fading and Middleton's Class-A impulsive noise channels, and compare them with the BER performances of the corresponding statistical decoding algorithms for the three codes and channels. In all cases the performance over the AWGN channel of the non-statistical algorithms is almost the same or slightly better than that of the statistical algorithms. In some cases the performance over the two non-Gaussian channels of the non-statistical algorithms is worse than that of the statistical algorithms, but the use of a simple signal amplitude limiter placed before the decoder input significantly improves the actual and relative performances of the algorithms. Thus there is no performance loss, and sometimes a significant performance gain, for the proposed decoding algorithms. A major advantage of our algorithms is that estimation of the channel signal-to-noise ratio is not required, which in practice simplifies system implementation. In addition, we have found that the processing complexity of the non-statistical algorithms is similar or slightly less than that of the corresponding statistical algorithms, and is significantly less for the LDPC codes over all of the channels.

KEYWORDS

Channel coding, error correction codes, decoding, communication channels

1. INTRODUCTION

Efficient error control codes like low-density parity-check (LDPC) codes [1], turbo (convolutional or block) codes [2] and polar codes [3]have been specified for use in most modern communications systems standards. This is because these codes have powerful decoding algorithms which achieve excellent bit error rate (BER) performances, making them the best option for communication over highly noisy channels.

LDPC codes are extensively used in many digital television standards [4], [5], all specifying use of a combination of LDPC and Bose–Chaudhuri–Hocquenghem (BCH) codes as the error-correcting technique. They are also extensively used in several standards for Local Area Networks (LAN) and Metropolitan Area Networks (MAN) [6]-[10]. Similar standards established in China also make use of LDPC codes. The Consultative Committee for Space Data systems, (CCSDS) specifies several standards for governmental agencies [11].

DOI :10.5121/ijcsea.2019.9502

Turbo codes are used in 3G and 4G mobile telephony standards [12], and in the standard for WIMAX, incorporating block and convolutional turbo coding.

The relatively new family of polar codes are increasingly being specified in modern communication systems standards. They are being analysed as an alternative coding technique for 5G wireless communications[13],[14]. Several researchers have evaluated the possible advantages of using polar instead of LDPC codes for wireless transmission at 60 GHz.

The decoders of all these error-correcting codes input soft (real-number) values from the channel, which are then processed by means of a soft decoding algorithm to produce soft values that estimate the confidence of each individual bit (or symbol, in general) output from the decoder. So these are soft-input, soft-output (SISO) decoders. The soft outputs can then be further refined, as in an iterative decoder, or used as inputs to another decoder, as in a concatenated scheme [15]. Gallager's sum-product (SP) decoding algorithm [1] for LDPC codes, also applied by Tanner to codes defined on bipartite graphs [16], and by Wiberg, et al, to general graphs [17]; the Bahl, Cocke, Jelinek and Rajiv (BCJR) decoding algorithm [18] for convolutional and turbo codes [2]; and Arikan's non-iterative successive cancellation (SC) decoding algorithm for polar codes [3]; are all examples of SISO decoders.

In most cases the processing complexity of the algorithms can be significantly reduced by using calculations in the logarithmic domain. All of these algorithms are essentially forms of statistical belief-propagation decoding [19] and, importantly, they rely on knowledge of the variance of the channel noise distribution. In practice this means that the Eb/No ratio or the signal-to-noise ratio (SNR) must be known or estimated, since the performance of the algorithm deteriorates if the estimate is inaccurate [20]. Some degradation of BER performance is accepted in a trade-off with complexity, as in the case of the min-sum approximation of the sum-product algorithm, and the max-LogMAP approximation of the BCJR turbo algorithm [21], [22].

In this paper we describe novel non-statistical SISO decoding algorithms which do not require knowledge or estimation of the variance of the channel noise. These algorithms make use of the existing decoding structures as mentioned above, such as the SP, BCJR, and SC algorithm structures, but the two novel aspects of our non-statistical algorithms are that the decoding metric is squared Euclidean distance, instead of a statistical metric; and that antilog-log (AL) sums are used to process the metrics, as explained in Section 2 below. Our algorithms also benefit from processing in the log domain. They can be used with any other decoding structure, such as the min-sum and max-LogMAP structures previously mentioned, delivering the same performance and complexity advantages, without the need to approximate the channel variance and perform scaling operations. Codes having any combination of parameters (length, distance, rate, etc.) can be used, and operation over a broad range of channel characteristics is possible, including fading and impulse-noise channels. All of these properties combine to make our non-statistical algorithms universally applicable. As we show in Sections 3, 4 and 5 below, the BER performances of our algorithms are as good as or slightly better than those of the corresponding statistical algorithms, confirming initial results [23], [24], and showing near-maximum-likelihood performances. In some cases, the algorithms offer a reduction in decoding complexity, which is an advantage for the practical implementation of decoders for powerful and effective codes, such as those in the standards mentioned above. Our comparisons assume the use of BPSK modulation over the channel, but the algorithms can also be used with higher order modulation schemes, such as MQAM and MPSK.

The rest of this paper is organised as follows. Section *2* describes the basic non-statistical SISO algorithm, illustrated by means of simple examples. Section *3* is devoted to the application and comparative simulation performance of our algorithms when decoding LDPC, turbo and polar codes over the additive white Gaussian noise (AWGN) channel. Section *4* and *5* do the same for

the fading and impulse noise channels, respectively. Section *6* discusses the results and concludes the paper.

2. THE BASIC IDEA AND SIMPLE EXAMPLES

In this section the basic concept of the non-statistical soft-decision decoding algorithm is described, and its application is explained by means of some simple examples.

It is assumed that binary data is transmitted over an additive white Gaussian noise (AWGN) channel, using signals of amplitude ± 1 when sampled after noiseless demodulation. The data is encoded using an (n, k, d) binary linear block code C, where n is the block length, k the dimension (number of information bits) and d the Hamming distance.

Let the noisy, real number received values at the input to the decoder be y_i, $i = 1,2, \ldots n$. The hard-decision estimate of a received bit is 0 if $y_i \geq 0$, or 1 if $y_i < 0$. These estimates can then be used in a hard-decision (HD) decoder for the code C.

To avoid the loss of information inherent in hard-decision estimates, compute the Euclidean distances between the received value and ± 1, the ideal received values in the absence of noise and interference on the channel. These distances are:

$$d_{i0} = |y_i - 1| d_{i1} = |y_i + 1| \tag{1}$$

Note that if $d_{i0} < d_{i1}$, then the hard-decision estimate of y_i is a 0, and vice-versa.
Normally the Euclidean distances are squared to reflect their energy, which then avoids computing the modulus (absolute value). Thus:

$$d_{i0}^2 = (y_i - 1)^2 d_{i1}^2 = (y_i + 1)^2 \tag{2}$$

As is well known, these distances obey the triangle inequality, and can now be used in a soft-decision (SD) decoder to determine the most likely transmitted codeword. A very simple example makes this clear.

Let the code C be the binary $(n, k, d) = (3,2,2)$ single parity check (SPC) code. The codewords in this code are $000, 101, 011$and 110. Let us assume that the received values are $y_i = -0.8, 0.3, 1.1$for $i = 1,2,3$. The hard-decision estimates are therefore 100. These do not form a codeword, so at least one error is present, which cannot be corrected by using only the HD estimates.

The squared Euclidean distances are:

$$d_{i0}^2 = \{3.24,\ 0.49,\ 0.01\} \tag{3}$$

$$d_{i1}^2 = \{0.04,\ 1.69,\ 4.41\} \tag{4}$$

These sums indicate that the most likely transmitted codewordis 110, as it has the smallest sum. In other words, 110 is the maximum likelihood (ML) codeword for this set of input values to the SD decoder for this SPC code. Thus a hard-decision error in the second position has been corrected to a 1, confirming the enhanced performance of a SD decoder over that of a HD decoder.

Table 1. Expanded table of the elements of the squared distance sums

$\mathbf{d_{i0}^2}$	$\mathbf{d_{i1}^2}$
$d_{10}^2(000) = 3.24 + (0.49 + 0.01) = 3.74$	$d_{11}^2(101) = 0.04 + (0.49 + 4.41) = 4.94$
$d_{10}^2(011) = 3.24 + (1.69 + 4.41) = 9.34$	$d_{11}^2(110) = 0.04 + (1.69 + 0.01) = 1.74$
$d_{20}^2(000) = 0.49 + (3.24 + 0.01) = 3.74$	$d_{21}^2(011) = 1.69 + (3.24 + 4.41) = 9.34$
$d_{20}^2(101) = 0.49 + (0.04 + 4.41) = 4.94$	$d_{21}^2(110) = 1.69 + (0.04 + 0.01) = 1.74$
$d_{30}^2(000) = 0.01 + (3.24 + 0.49) = 3.74$	$d_{31}^2(101) = 4.41 + (0.04 + 0.49) = 4.94$
$d_{30}^2(110) = 0.01 + (0.04 + 1.69) = 1.74$	$d_{31}^2(011) = 4.41 + (3.24 + 1.69) = 9.34$

If the code is part of an error-control coding scheme that makes use of SISO decoding, soft estimates or confidence values for the decoded bits are needed. The decoder would then become a SISO decoder, equivalent to an a posteriori probability (APP) decoder.

Table 1 is an expanded table of the elements of the squared distance sums. The rows of the table contain the two sums which correspond to a 0 or a 1 in the first, second or third positions in the four codewords, as indicated by the subscripts and codeword arguments of d_{i0}^2 and d_{i1}^2. Within each minimum distance sum, the first element is the intrinsic squared distance, and the two elements in brackets are the two extrinsic squared distances corresponding to that particular codeword, thus highlighting their contribution to the parity check involving the intrinsic element.

To obtain a good soft estimate for a 0 or a 1 in each of the three positions it is necessary to take into account both of the corresponding distance sums. As is well known, in the case of an AWGN channel an estimate can be obtained by calculating the likelihoods of each sum, adding them, and converting the resulting probability sum back to a combined soft distance estimate. For example, in the case of a soft estimate for a 0 in the second position, the calculations would be:

$$p_{20}(000) + p_{20}(101) = \frac{1}{\sqrt{2\pi}\sigma} e^{-d_{20}^2(000)/(2\sigma^2)} + \frac{1}{\sqrt{2\pi}\sigma} e^{-d_{20}^2(101)/(2\sigma^2)} \tag{5}$$

$$C_{20}(000,101) = -\ln[p_{20}(000) + p_{20}(101)] \tag{6}$$

whereC_{20} is the combined soft squared distance estimate for a 0 in the second position, and σ is the noise variance.

However, the aim here is to devise useful non-statistical combined estimates, so as to avoid the need to know the noise variance. This can be done by modifying equations (5) and (6) as follows:

$$p_{20}(000) + p_{20}(101) = b^{-d_{20}^2(000)} + b^{-d_{20}^2(101)} \tag{7}$$

$$D_{20}(000,101) = -\log_b[p_{20}(000) + p_{20}(101)] \tag{8}$$

Where b is a suitable base value. The left-hand terms in (7) are now pseudo-likelihoods, and D_{20} in (8) is the combined estimate for a 0 in the second position. The estimate for a 1 in the second position can be similarly obtained. Thus, setting $b = 2$, the estimate calculations for the second position in the codeword are as follows:

$$-\log_2(2^{-3.74} + 2^{-4.94}) = -\log_2(0.075 + 0.033) = 3.21 \qquad \text{(for a 0)} \tag{9}$$

$$-\log_2(2^{-9.34} + 2^{-1.74}) = -\log_2(0.002 + 0.299) = 1.73 \qquad \text{(for a 1)} \tag{10}$$

which confirms that the estimates for the second decoded bit indicate that it is a 1, as found previously. By comparing these modified combined estimates with the individual estimates in the third and fourth rows of Table *1*, it can be seen that the combined estimates are each less than the smaller of the two individual estimates. This is correct, because the smaller of the two sums is already a first soft estimate of the decoded bit, and its combination with the other estimate can only improve the combined soft estimate, by reducing its value.

It can be seen that effectively the calculation for the combined estimate consists of taking the antilogs of the two negative individual estimates and then taking the negative log of the addition of the antilogs, all to a suitable base, 2 in this example. This simple AL process will be called the basic-AL algorithm. The combined estimates, after suitable scaling, can be carried forward for use in the second decoding operation of a concatenated or iterative scheme. In this simple example, a suitable scaling would be to add the two estimates, divide each by the total, and then multiply each by 2 (as 2 is the distance between the ideal received values, ± 1). For the 0 estimate in the second position, 3.21becomes1.30; and for the 1 estimate, 1.73 becomes 0.70. Clearly the scaled estimates indicate that the HD decoded bit should be a 1, but also that it is a low confidence 1, as the SD values are only 0.3 away from the HD zero threshold. Scaling has the additional advantage of providing a single value from which the 0 and 1 estimates can be calculated. Table *2* lists the basic-AL combined and scaled estimates for all the decoded bits in the simple example.

Table 2. Expanded table of the elements of the squared distance sums

position	for	Antilog-sum	scaled	SD result	HD result	confidence
1	0	3.70	1.40	-0.4	1	fairly low
1	1	1.59	0.60			
2	0	3.21	1.30	-0.3	1	low
2	1	1.73	0.70			
3	0	1.42	0.45	0.55	0	medium
3	1	4.84	1.55			

It is not surprising that the decoded bits are not very confident in this very simple SPC code example. The important thing is that the AL combining process produces numerically reliable decoded estimates for further use.

In general, if the values (distance sums) to be combined are $A, B, C, \ldots$, then the result is given by:

$$D(A,B,C,\ldots) = -\log_b\left(b^{-A} + b^{-B} + b^{-C} + \cdots\right) \tag{11}$$

where$b$ is any convenient base for the logarithm. It follows that:

$$\begin{aligned} D(B,A) &= D(A,B) \\ D(A,A) &= A - \log_b 2 \\ D(A,B) &\leq \min(A,B) \\ D(A,B) &\approx A \text{ if } B \gg A \\ D(A,B) &= \min(A,B) - \log_2\left(1 + 2^{|A-B|}\right) \end{aligned} \tag{12}$$

The basic-AL algorithm as used in the above simple example would be impractical once the number of codewords in the code becomes large. However, the AL concept can be applied to any practical decoding structure, such as the Tanner (factor) graph or the trellis of a code [24], [25]. Fig.*1* shows the graph of the (3,2,2) SPC example code, where the variable nodes 1, 2 and 3 are linked by edges to a constraint node (the parity check). Using this graph and the AL process,

reliable estimates of the three decoded bits can be obtained. For example, if the received values on the edges from nodes 1 and 3 towards the constraint node are $\{d_{10}^2, d_{11}^2\}$ and $\{d_{30}^2, d_{31}^2\}$ respectively, then the values passed from the constraint node towards node 2 are given by:

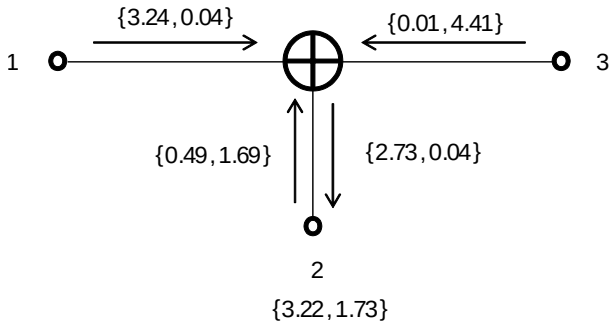

Figure 1. Graph of the (3,2,2) SPC example code.

$$D_0 = D[(d_{10}^2 + d_{30}^2), (d_{11}^2 + d_{31}^2)]$$
$$D_1 = D[(d_{10}^2 + d_{31}^2), (d_{11}^2 + d_{30}^2)]$$

The 0 and 1 estimates at node 2 are then given by $D_0 + d_{20}^2$ and $D_1 + d_{21}^2$.

Inserting the example values (shown in Fig. 1) then gives the estimated values are $\{3.22, 1.73\}$. Allowing for small rounding errors, these are the same as those previously calculated. The decoded estimates at nodes 1 and 3 can be similarly calculated, and are again the same as previously found. Clearly, the form of this process using the AL calculations on the graph of the code has the same structure as that of the well-known SP decoding algorithm [21], [25] with the crucial distinction that the metric used is the AL metric instead of the Likelihood Ratio (LR) or Log Likelihood Ratio (LLR) metrics of the SP algorithm. This will be denoted the AL-SP algorithm.

Figure 2 shows the trellis of the SPC code example. Decoding from left to right, the values on each edge are the squared distances, initial in the first stage and cumulative in the second and third stages.

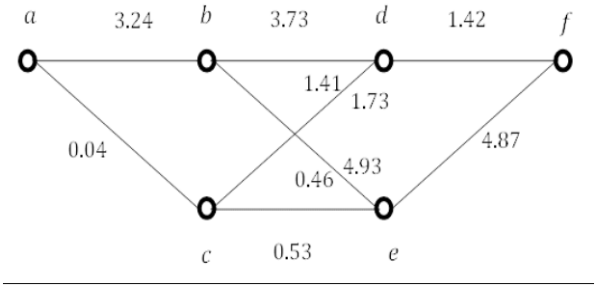

Figure 2.Trellis of the SPC code example.

In the second stage, two values arrive at each of the nodes d and e, and so need to be combined. At node d, $D(3.73,1.73) = 1.41$, and at node e, $D(4.93,0.53) = 0.46$. The 0 and 1 estimates at the final stage (position 3) are therefore 1.42 and 4.87 respectively, which are the same as those found previously (allowing for rounding errors). Decoding from right to left, and combining values as required, then 0 and 1 soft estimates for positions 1 and 2 can be obtained, which will be the same as before. This forward-backward decoding process using the trellis of the code and

the AL metric, has the same structure as that of the BCJR algorithm [18], and will be denoted the AL-BCJR algorithm. In a similar way, the AL metric can be used in a modified form of Arikan's SC decoding algorithm [3], and will be denoted the AL-SC algorithm. As before, the processing structure of the AL-SC algorithm is the same as that of the SC algorithm, but uses the AL metric, as will be described in Section *3.3* below.

So far we have used $b = 2$ as the base of the logarithms in the AL algorithms, but now we will discuss further the choice of a value for b. In a previous work [23], we found that the value $b = 2$ was suitable for codes of rate $1/2$. However, for higher-rate codes it was necessary to increase the value of b to 3 or more in order to maintain near-optimum BER performance. This was because the increasing number of edges in the Tanner graph of the higher-rate codes led to a form of computational instability when $b = 2$, which disappeared when the value of b was raised. In a deeper study, we were able to determine the optimum value of b for a range of code parameters (rate, length, parity-check matrix density, etc.) and channel Eb/No values. As this study was published in Spanish, we will briefly translate the main results and conclusions of paper [26].
In a simulation done for the BER performance of a $(273,171)$ LDPC code with rate 0.626 as Eb/No and b are varied, using the AL-SP decoding algorithm, we have found that values from2.5to8 give almost the same performance.

Figure 3 is the BER as a function of parameter b for a $(273,171)$ LDPC code at$Eb/No = 4.4dB$ that shows that the optimum value of $b$is 3.5, and explains why b values of 2.5 and 8 provide near-optimum results. As shown in [26], simulation of the BER performance of$(120,56)$ LDPC code with rate 0.467,for different values of Eb/No,shows that the optimum value of b is very close to 2, virtually independent of Eb/No. We found that this feature persists for a wide range of code parameters, as shown in Table 3, where the optimum value of b varies between 1.7 and 3.45, and is almost independent of the SNR on the channel. Overall, we found that very good performance can be obtained by selecting $b = 2$ for codes with $R_c \leq 1/2$, and $b = 3$ or $b = 4$ for $R_c \geq 1/2$. This dependency to the code ratealso occurs when iterative MAP decoding is implemented at a fixed value of Eb/No to avoid estimating noise dispersion[27].

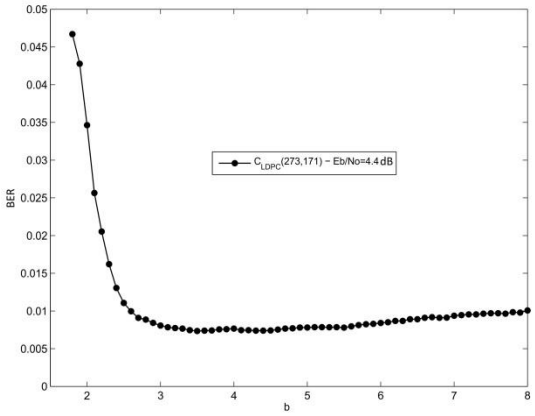

Figure 3.BER Performance of a (273, 171) LDPC code as a function of b, $Eb/No = 4.4dB$.

The AL algorithms are of course non-statistical, as they do not require knowledge of the channel Eb/No or error probability. However, important questions remain: what is the performance of the AL decoding algorithms, and how do they compare with the performance of the widely used corresponding statistical decoding algorithms? In the following sections it is shown that the performances of the non-statistical AL algorithms are almost the same as those of the corresponding statistical algorithms, but sometimes significantly better. It is also shown that in some cases the AL algorithms are less complex to implement.

Table 3. Optimum value of parameter b for different LDPC codes

Code length	Code Rate	Num '1s'	Optimum b
60-30	0.5	3	2.80
96-32	0.3	2, 3 (2.67)	1.70
96-32	0.3	3 (2.67)	1.70
120-56	0.46	3	2
204-102	0.5	3	2.2
204-102	0.5	5	2.6
271-144	0.53	3	2.5
273-191	0.69	3	3.45
408-204	0.5	3	1.95
1008-504	0.5	3	2

3. THE AL DECODING ALGORITHMS FOR LDPC, TURBO AND POLAR CODES OVER THE AWGN CHANNEL

In this Section we describe the details of the non-statistical AL decoding algorithms for the three classes of codes. We present simulation results of the BER performance of these codes when used over the AWGN channel, and compare them with the BER performance of the corresponding statistical decoding algorithms.In the case of the AL decoding algorithms, initialization consists of the calculation of the squared distance values in terms of the received real number values from the channel, as previously set out in (1) and (2) in Section 2:

$$d_i^2(0) = (y_i - 1)^2 \quad d_i^2(1) = (y_i + 1)^2 \tag{13}$$

For the corresponding statistical algorithms, initialization consists of calculating the following error probabilities:

$$P_i(0) = \frac{1}{\sqrt{2\pi}\sigma}\, e^{-\frac{Eb}{2\sigma^2}(y_i-1)^2} \quad P_i(1) = \frac{1}{\sqrt{2\pi}\sigma}\, e^{-\frac{Eb}{2\sigma^2}(y_i+1)^2} \tag{14}$$

where σ is the AWGN channel noise variance, from which LRs or LLRs can be determined.

3.1. The LogAL-SP Decoding Algorithm for LDPC Codes in AWGN

LDPC codes are decoded by using an iterative SP algorithm, or its logarithmic version, the LogSP decoding algorithm [21], [25]. The LogSP decoding algorithm uses logarithmic operations which reduce the decoding complexity of the SP algorithm, and this advantage also extends to the LogAL-SP algorithm, where the statistical information calculations of the LogSP algorithm are replaced by AL calculations using squared soft distance values. As above in Section 2, initialization of the logAL-SP algorithm consists on setting coefficients $q_{ji}(0)$and $q_{ji}(1)$, $i = 1,2,\dots,n$, $j = 1,2,\dots,m$, to the corresponding squared distance values at each symbol node:

$$q_{ji}(0) = d_i^2(0) \quad \text{and} \quad q_{ji}(1) = d_i^2(1) \tag{15}$$

A detailed derivation of expressions for the simplified LogAL-SP is presented in [23]. A simplification of calculations in the horizontal step is as follows:

$$a_{ij} = f_+(q_{ji}) \text{and} b_{ij} = -f_-(q_{ji}) \tag{16}$$

where: $f_{-}(q_{ji}) = \left|log_b\left(1 - b^{-|q_{ji}(0)-,q_{ji}(1)|}\right)\right|$and $f_{+}(q_{ji}) = \left|log_b\left(1 + b^{-|q_{ji}(0)-,q_{ji}(1)|}\right)\right|$

For the logAL-SP algorithm, simulations were done by setting the base of the logarithmic calculations to $b = 2$, which gives excellent results for low or medium code rates. However, for high rate codes better results can be obtained with $b = 4$ or more [23].

A simulation of the BER performance of a (4096, 2048) LDPC code [29] in AWGN is shown in Fig. 4, where it can be seen that the curves of the LogAL-SP and the LogSP decoding algorithms are almost identical. These new results were obtained by transmitting 2500 encoded messages, decoded with $b = 2$ and 16 iterations. In this and all the following performance graphs in the paper, each point on the curves represents the occurrence of at least 100 errors in the whole transmission.

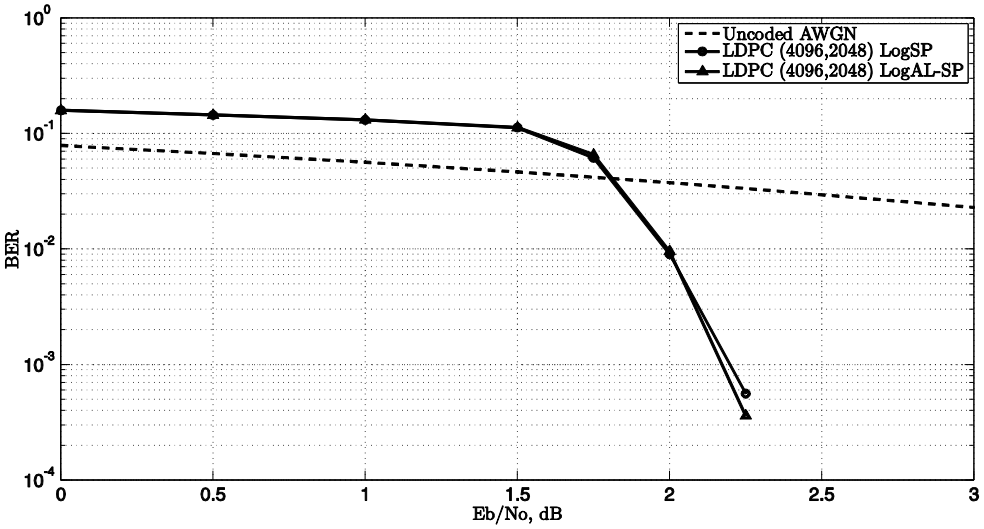

Figure 4.BER of the (4096, 2048) LDPC code, AWGN channel, LogAL-SP and LogSP decoding algorithms.

A complexity analysis and the corresponding Field Programmable Gate Array (FPGA) implementation has been done in [24] for the LogAL-SP and for the classic LogSP algorithm. This comparison shows that the LogAL-SP decoding algorithm is approximately 8% less complex than the LogSP. This complexity reduction was also verified via FPGA implementations; the LogAL-SP decoder was faster and used smaller numbers of logic components and registers.

3.2. The LogAL-BCJR Decoding Algorithm for Turbo Codes in AWGN

Turbo codes [2] are decoded by using the statistical LogBCJR decoding algorithm, and the results from each constituent decoder are then iteratively exchanged a sufficient number of times to achieve an enhanced final result. This is the turbo decoding algorithm [21], [25]. In this section the constituent codes are decoded using the Euclidean distance metric in the non-statistical AL-BCJR algorithm, as the simple example in Section *2* shows, and then the results from the constituent decoders can be iteratively exchanged as in the turbo decoding process. As above in Section *2*, for polar format binary signals we can obtain conditional LLRs of the form [30]:

$$LLR = ln\left(\frac{e^{\frac{-E_b}{2s^2}(y_i-1)^2}}{e^{\frac{-E_b}{2s^2}(y_i+1)^2}}\right) = \frac{-E_b}{2s^2}(y_i-1)^2 + \frac{E_b}{2s^2}(y_i+1)^2 = 2\frac{E_b}{s^2}y_i = L_c y_i \quad (17)$$

Thus, the conditional LLR for an AWGN channel is proportional to the value of the sample of the optimally received signal y_i, and the constant of proportion is $L_c = \frac{2Eb}{\sigma^2}$, a measure of the signal-to-noise ratio in the channel. The LLRs in the BCJR decoding algorithm for decoding turbo codes are [21], [25], [30]:

$$LLR = L(b_i / Y) = \ln\left(\frac{\sum_{\{u',u\}\to b_i=+1} a_{i-1}(u')\gamma_i(u',u)b_i(u)}{\sum_{\{u',u\}\to b_i=-1} a_{i-1}(u')\gamma_i(u',u)b_i(u)}\right) \quad (18)$$

where $Y = Y_1^n = \{Y_1, Y_2, \ldots, Y_n\}$ is a sequence of n received symbols, u and u' are possible values that trellis states can adopt. The term $\alpha_{i-1}(u')$ represents the forward information calculated over the corresponding trellis in the branch defined by $\{u', u\}$; $\gamma_i(u', u)$ represents the present information at branch $\{u', u\}$, and $\beta_i(u)$ is the future information calculated at that branch. Details of the BCJR decoding algorithm and its logarithmic version can be seen in [21], [25], [30].

In (18) b_i is the message bit $b_i = \pm 1$, and $L(b_i)$ is the log Likelihood ratio:

$$L(b_i) = ln\frac{(P(b_i=+1))}{(P(b_i=-1))} \quad (19)$$

The corresponding logarithmic expression of $G(u', u) = ln(\gamma_i(u', u))$ is [30], [21],[25]:

$$G(u', u) = \ln(\gamma_i(u', u)) = b_i L(b_i)/2 + \frac{L_c}{2}\sum_{k=1}^{m} y_{ik}\, x_{ik} \quad (20)$$

The AL operation can be applied to these coefficients to lead to equivalent expressions, where base $b = 2$ is used instead of $b = e$, and the parameter $L_c = \frac{2Eb}{\sigma^2}$ does not have to be estimated, but instead adopts the fixed value $L_c = 2$, which we found experimentally to be effective. Defining $G(u', u) = ln(\gamma_i(u', u))$, $A_{i-1}(u') = ln(\alpha_{i-1}(u'))$ and $B_i(u) = ln(\beta_i(u))$ in the case of logarithmic operation, we convert products of the form $\alpha_{i-1}(u')\gamma_i(u', u)\beta_i(u)$ into the sum $A_{i-1}(u') + G(u', u) + B_i(u)$. The same happens in AL operation with the only difference being that the base $b = 2$ and L_c is a constant. Therefore AL operations are equivalent to logarithmic operations in the statistical BCJR decoding algorithm. This means that the LogAL-BCJR and the LogBCJR decoding algorithms have the same processing complexity, but the initialisation complexity of the LogAL-BCJR algorithm is lower.

A simulation of the BER performances of the LogAL-BCJR and the LogBCJR algorithms, for a binary turbo code using as constituent encoders two $1/2$-rate binary systematic recursive convolutional (111,101) encoders operating over the AWGN channel, is shown in Fig. 5. The interleaver is a random interleaver of size $N = 2000$, and puncturing is applied so that the final rate of the whole turbo code is $1/2$. The simulation has been done by transmitting 25000 messages of size 2000 bits each.

3.3. The LogAL-SC Decoding Algorithm for Polar Codes in AWGN

Polar codes [3] are typically decoded by using the SC decoding algorithm. We will apply the AL decoding algorithm to the processing structure of the logarithmic SC (LogSC) decoding algorithm to obtain the non-statistical LogAL-SC decoding algorithm. In its usual form, and following the notation as in [3], [32], the SC algorithm starts by receiving the samples vector $y = (y_1\ y_2\ \dots\ y_n)$, $i = 1,2,\dots,n$, to obtain statistical LLR values $L_{n,i}$ that are used to determine the input bit estimations by recursively applying the following expressions:

$$L_{j,i} = \begin{cases} f\left(L_{j+1,i}; L_{j+1,i+2^j}\right) = f(a,b) & \text{if } B(j,i) = 0 \\ g\left(\hat{s}_{j,i-2^j}; L_{j+1,i-2^j}; L_{j+1,i}\right) = g(\hat{s},a,b) & \text{if } B(j,i) = 1 \end{cases} \quad (21)$$

where$\hat{s}$ is a modulo-2 partial sum of decoded bits, and the parameter $B(j,i)$ is defined as $B(j,i) \triangleq \frac{i}{2^j} mod\ 2$. Indexes are $0 \le j < n$, and $0 \le i < N$.Functions f and g in (21) are calculated using the following expressions:

$$L_{j,i} = \begin{cases} f(a,b) & = & \frac{1+ab}{a+b} \\ g(\hat{s},a,b) & = & a^{1-2\hat{s}}b \end{cases} \quad (22)$$

Applying logarithmic simplification of expressions (21) and (22), by setting $a = e^{\lambda_a}$, $b = e^{\lambda_b}$, the following expressions are obtained:

$$L_{j,i} = \begin{cases} f(\lambda_a,\lambda_b) \simeq sign(\lambda_a) sign(\lambda_b) mín(|\lambda_a|,|\lambda_b|) \\ g(\hat{s},\lambda_a,\lambda_b) = (-1)^{\hat{s}}\lambda_a + \lambda_b \end{cases} \quad (23)$$

The LogAL-SC decoding algorithm is initialized by taking the difference between the squared distances at the decoder input with respect to the two possible ideal values $+1$ and -1. From (2) the result is equal to $-4y_i$ that is, the initialization values are directly scaled values of the channel samples. Applying the AL procedure leads to [33]:

$$L_{j,i} = \begin{cases} f(\lambda_a,\lambda_b) \simeq máx(0,\lambda_a + \lambda_b) - máx(\lambda_a,\lambda_b) \\ g(\hat{s},\lambda_a,\lambda_b) = (1-2\hat{s})\lambda_a + \lambda_b \end{cases} \quad (24)$$

Expressions (24) are equivalent to expressions (23), therefore in the case of the LogAL-SC decoding algorithm there is no reduction in decoding complexity when compared to that of the LogSC. The advantage of the use of the LogAL-SC algorithm with respect to the LogSC decoding algorithm is that the initialization step is simpler and does not require knowledge of the variance of the noise present on the channel. Simulations of the LogAL-SC and LogSC algorithm BER performances in AWGN of the (4096, 2048) and the (4096, 3072) polar codes are shown in Fig. 6, obtained by transmitting 50,000 messages of size 2048 and 10,000 messages of 3072 bits respectively. Once again, the performance of the non-statistical LogAL-SC decoding algorithm is very close to that of the LogSC decoding algorithm, confirming previous results for the (256,128) polar code [33].

4. THE AL DECODING ALGORITHMS FOR LDPC, TURBO AND POLAR CODES OVER THE RAYLEIGH FADING CHANNEL

In this Section we present the results of BER simulations of the non-statistical AL decoding algorithms for the three code classes when used over the Rayleigh fading channel, and compare them with the results for the corresponding statistical algorithms. In order to make the comparisons as fair as possible, we do not make use of any additional channel state information (CSI) when decoding with the statistical algorithms, but use only the channel noise variance.

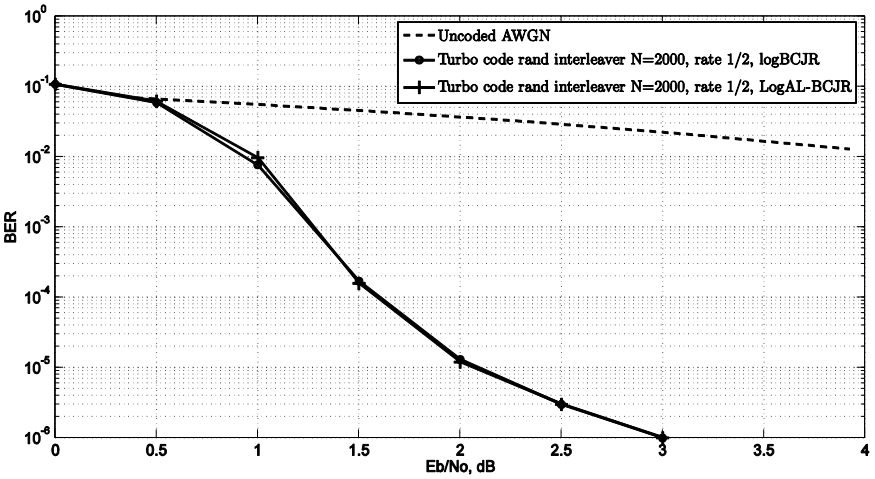

Figure 5.BER in AWGN of a $1/2$-rate turbo code, LogAL-BCJR and LogBCJR algorithms.

The Rayleigh fading channel model characterises wireless transmissions where the signal arrives at the receiver over multiple propagation paths, and an attenuation factor $\alpha_n(t)$and also a time delay $\tau_n(t)$can be associated with each one of them. Both parameters are time dependent because in general terms the channel is time varying.

The Rayleigh fading channel is moldelled as with a time varying baseband impulse response $\alpha(t) = \sum_n \alpha_n(t)e^{-\theta_n(t)}$, where $\theta_n(t) = 2\pi f_c \tau_n(t)$ is the phase of the $n-th$ path, which is interpreted as a circularly symmetric complex Gaussian random variable that is of the form $= \alpha_{Re} + j\alpha_{Im}$[35], where the real and imaginary parts are zero mean independent and identically distributed Gaussian random variables. The corresponding probability density function (pdf) is:

$$p_R(\alpha) = \frac{\alpha}{\sigma^2} e^{-\alpha^2/(2\sigma^2)} \tag{25}$$

A more detailed description of this model can be found in [35].

For AL decoding algorithms initialization is implemented by using the following expressions:

$$d_i^2(1) = (y_i - a)^2; \quad d_i^2(0) = (y_i + a)^2; \quad d_i^2(1) - d_i^2(0) = -4ay_i \tag{26}$$

where the value of a is set to unity, after we found that decoding performance is not sensitive to values of a over quite a wide range.

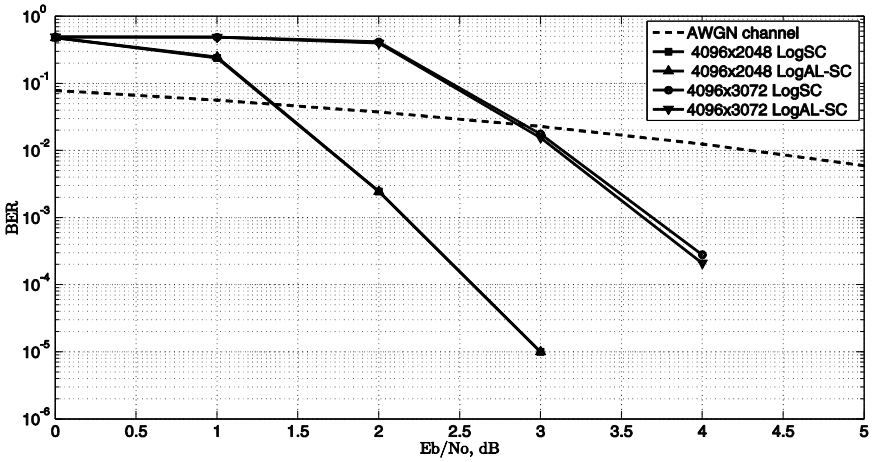

Figure 6.BER of the (4096, 2048) and the (4096, 3072) polar codes, AWGN channel, LogAL-SC and LogSC decoding algorithms.

The statistical decoding algorithms that make use of LLR values and operate over the Rayleigh fading channel determines their initialization values from the following expressions [30]:

$$P_i(1) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{E_b}{2\sigma^2}(y_i - a)^2}; \qquad P_i(0) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{E_b}{2\sigma^2}(y_i + a)^2} \tag{27}$$

where again a can be set to unity. Then, the initialization value for the statistical algorithms is given by:

$$LLR_{R,i} = \frac{2y_i}{\sigma^2} \tag{28}$$

Noting the very significant performance improvements obtained when using a simple signal amplitude limiter in the case of the impulsive noise channel [33], [36], we applied the same idea to the fading channel. The received signal samples were passed through a limiter with symmetrical limit values $A_{lim} = \pm 1$ before the input to the decoder, which resulted in significantly improved LDPC code performance when decoded using both the non-statistical and statistical algorithms, as shown next.

4.1. The LogAL-SP Decoding Algorithm for LDPC Codes in Rayleigh Fading

Figure 7 shows the performance of the (2560, 1280) LDPC code [29] over the Rayleigh fading channel using the LogAL-SP and LogSP decoding algorithms [23], [37]. Simulation of the code performance is done by transmitting 5000 message blocks of 1280 message bits each, with 16 decoding iterations.In Fig. 7, the left-hand pair of curves were obtained with the use of a signal amplitude limiter. As seen, if the limiter is not applied, the performance of both algorithms is very poor and LogAL-SP is much worse than LogSP. However, with the limiter there is a significant performance gain over the curves where the limiter is not applied, and the LogAL-SP algorithm outperforms the LogSP algorithm.

4.2. The LogAL-BCJR Decoding Algorithm for Turbo Codes in Rayleigh Fading

Figure 8 shows the BER performance of a $1/2$-rate turbo code formed from two $1/2$ -rate binary systematic recursive convolutional (111,101) encoders, with puncturing and a random interleaver of size $N = 2000$ decoded using the LogAL-BCJR and the LogBCJR algorithms over the

Rayleigh fading channel, both with and without a ± 1 limiter. The simulations involve transmitting 1000 messages of size 2000 bits each.In the case of the LogBCJR and the LogAL-BCJR algorithms there are no significant differences in the internal calculation procedure for both decoding algorithms. They only differ in the initialization step. Once again the use of the limiter greatly improves the performance of the BCJR code on this fading channel. Without the limiter, the performance of both the LogAL-BCJR and the LogBCJRalgorithms is similar but very poor; with the limiter, both algorithms also have similar performance but with gains of over 16 dB at a BER of 10^{-4}. The processing complexities of the two algorithms are the same, but initialization is simpler in the AL case.

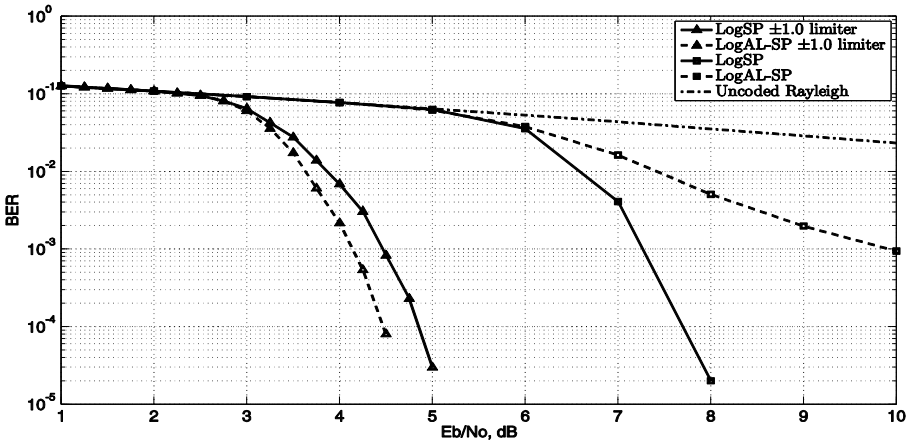

Figure 7.BER of a (2560, 1280) LDPC code, Rayleigh channel, LogAL-SP and LogSP algorithms.

4.3. The LogAL-SC Decoding Algorithm for Polar Codes in Rayleigh Fading

Figure 9 compares the BER decoding performance of a (4096, 3072) polar code constructed as in [3], using the LogAL-SC algorithm, the LogAL-SC algorithm with a ± 1 signal amplitude limiter, and the LogSC algorithm. The simulations involved transmitting 10000 messages of size 3072 bits each. As seen in Fig. 9, the performances of the three decoding algorithms are very close, and the use of the signal amplitude limiter does not result in a significant difference in BER performance for this code operating over the Rayleigh fading channel. These new results confirm those obtained in a previous investigation [33].

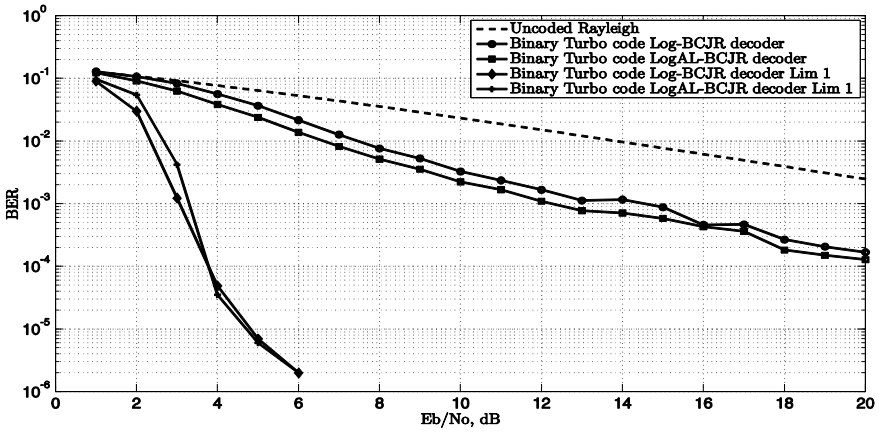

Figure 8.BER of a $1/2$-rate turbo code, LogAL-BCJR and LogBCJR algorithms, Rayleigh fading channel, with and without a limiter.

5. THE AL DECODING ALGORITHMS FOR LDPC, TURBO AND POLAR CODES OVER THE IMPULSIVE NOISE CHANNEL

In this Section the performances in impulsive noise (IN) of the non-statistical AL-based decoding algorithms for LDPC, turbo and polar codes are compared with those of the corresponding statistical decoding algorithms. The impulsive noise channel model used in our simulations is Middleton's Class- A impulsive noise channel [38]. As before, the statistical decoding algorithms are initialized using the channel noise variance σ, with no other CSI available at the receiver. As in the case of the fading channel simulations, we also apply a simple signal amplitude limiter at the input to the decoder, originally suggested in [36].

The channel probability density function (pdf) consists of a Poisson weighted sum of Gaussian distributions, given by:

$$P_n(y) = \sum_{m=0}^{\infty} \frac{A^m e^{-A}}{\sqrt{2\pi}\, m!/\sigma_m} e^{-y^2/(2\sigma_m^2)}, \sigma_m^2 = \sigma^2 \frac{\frac{m}{A}+\Gamma}{1+\Gamma}, \Gamma = \sigma_g^2/\sigma_i^2, \text{ and } \sigma^2 = \sigma_g^2 + \sigma_i^2 \qquad (29)$$

In the above expressions, m is the number of impulses over the estimation period; A is the average number of impulses during the period; Γ is the ratio between the background noise power σ_g^2 and the impulsive noise power σ_i^2; $\sigma_g^2$is the variance of the Poisson random variable m; and σ^2 is the total noise power. Both the AL and the statistical decoding algorithms are initialised using expressions (26) to (28) in Section *4*, with $a = 1$ the same as in the case of the Rayleigh fading channel. A more detailed description of this model can be found in [38].

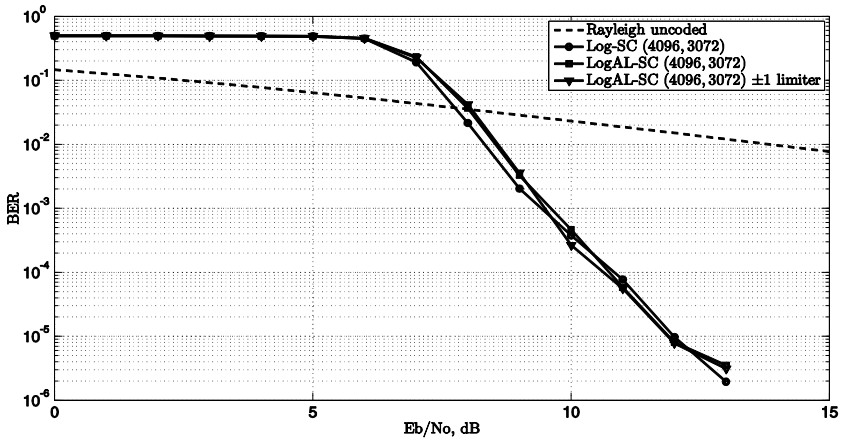

Figure 9.BER of a (4096, 3072) polar code, Rayleigh fading channel, LogAL-SC algorithm with and without a ± 1 signal amplitude limiter, and LogSC algorithm.

5.1. The LogAL-SP Decoding Algorithm for LDPC Codes in Impulsive Noise

A simulation of the performance of the long (4096, 2048) LDPC code, operating over Middleton's Class-A model of the impulsive noise channel, is presented. The parameters of the channel are $A = 0.1$, $\Gamma = 0.1$, and $M = 10$. As indicated above, we apply an input signal amplitude limiter of value $A_{lim} = \pm 1$ to improve the performance of both the LogAL-SC and the LogSC decoders. The simulations seen in Fig. 10have been done by transmitting 3000 messages of length 2048 bits each.

As seen in Fig. 10, the behaviour of the LogAL-SP decoder is worse than that of the LogSP decoder if no signal amplitude limiter is applied, but the performance of the LogAL-SP decoder

significantly outperforms that of the LogSP decoder when a ± 1 signal amplitude limiter is used. This coding gain of about 2dB is similar to that found in previous studies [37]. The decoding complexity is the same for both the LogAL-SC and the LogSC algorithms, but the former is simpler to initialize.

5.2. The LogAL-BCJR Decoding Algorithm for Turbo Codes in Impulsive Noise

Figure 11 shows the BER performance of a$1/2$-rate turbo code using two $1/2$-rate binary systematic recursive convolutional (111,101) encoders operating over Middleton's Class-A impulsive noise channel model, with parameters $A = 0.1$, $\Gamma = 0.1$ and $M = 10$. The simulations have been done by transmitting 1000 messages of size 2000 bits each after passing through a random interleaver of size $N = 2000$. The turbo code is decoded using the LogAL-BCJR and the LogBCJR algorithms, both with and without a± 1 amplitude limiter.

Figure 11 shows that once again the BER performance of both decoding algorithms is very significantly improved when the simple ± 1 signal amplitude limiter is used. With the limiter, the LogAL-BCJR decoder is slightly better than that of the LogBCJR decoder at low to medium values of Eb/No, but for higher values of this parameter, the performances become quite close to each other. An erratic floor effect is observed, caused by the fact that points below BER $< 10^{-5}$ are not statistically reliable since less than 100 errors are obtained. The processing complexity of both algorithms is same, but LogAL-BCJR is simpler to initialize.

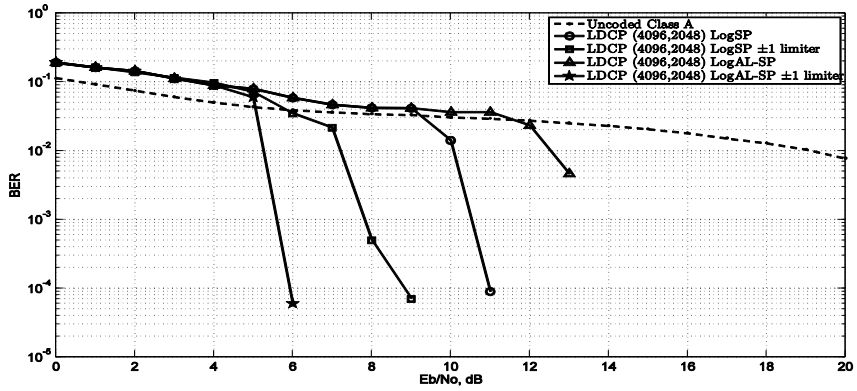

Figure 10.BER of the (4096, 2048) LDPC code, Middleton's Class-A channel, LogAL-SP and LogSP algorithms, with and without a ± 1 limiter.

5.3. The LogAL-SC Decoding Algorithm for Polar Codes in Impulsive Noise

A simulation of the performance of the large size (4096, 2048) polar code, operating over Middleton's Class-A impulsive noise channel model, is shown in Fig. 12. The parameters of the channel are $A = 0.1$, $\Gamma = 0.1$ and $M = 10$. Again we apply a ± 1 signal amplitude limiter to improve the performance of both the LogAL-SC and the LogSC decoders. The simulations were done by transmitting 2000 messages of size 2048 bits each.As seen in Fig. 12, the BER performances of the decoding algorithms are virtually the same, and the effect of the signal amplitude limiter is very significant. Both algorithms have the same processing complexity, but initialization is simpler for the LogAL-SC decoder.

6. CONCLUSION

In this paper we have presented a novel modification of the classic statistical SISO decoding algorithms for error-control codes. The proposed modification is based on two different aspects: the first is to replace statistical values by estimates based on squared Euclidean distance; the second is to implement the calculations of algorithms by means of antilog-sum (AL) operations. At the end of the process, soft estimates for each symbol in the received word are delivered at the output of the SISO decoder. Our universal SISO algorithm can be used to decode any particular error-correcting code, as the AL process can be applied to any convenient decoding structure for that code. For our purpose we have chosen three particular classes of codes: LDPC codes and the SP iterative decoding structure; turbo trellis codes and the BCJR two-way decoding structure; and polar codes with the SC decoding structure.

In all cases the decoding complexity is reduced by using logarithmic processing, so the three classes of codes and their decoding algorithm are labelled LogAL-SP, LogAL-BCJR and LogAL-SC. In addition, we have been able to refine the mathematical processes in the LogAL-SP case, which yields a further simplification of this decoding algorithm. By means of simulations, we have compared the BER performance of these non-statistical decoders with the BER performance of the corresponding statistical decoders for these codes, labelled LogSP, LogBCJR and LogSC, for three channel models: the AWGN channel, the Rayleigh fading channel, and Middleton's Class-A impulsive noise channel.

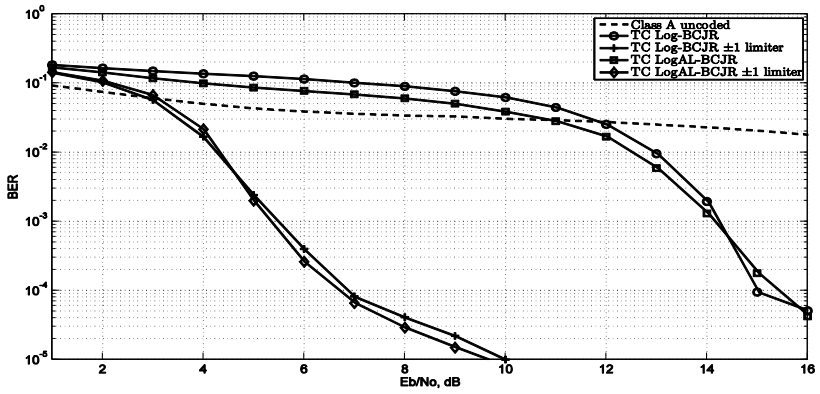

Figure 11.BER of a $1/2$-rate turbo code (111,101), Middleton's Class-A channel, LogBCJRand LogAL-BCJR algorithms, with and without the use of a limiter.

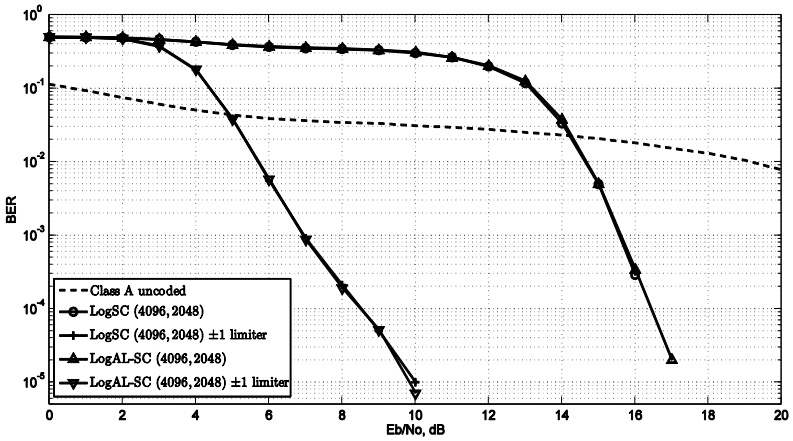

Figure 12.BER of the (4096, 2048) polar code, Middleton's Class-A channel, LogAL-SC and LogSC algorithms, with and without a ± 1 signal limiter.

The results of our simulations for the AWGN channel show that the BER performances of the three code classes are almost identical; there is no loss of performance when using the non-statistical decoding algorithms. In the case of the Rayleigh fading channel, the LDPC results were initially disappointing: non-statistical performance was worse than statistical performance, and both were poor. However, once a simple signal amplitude limiter is used at the input to the decoders, the situation is reversed: non-statistical is the same or better than statistical, and both are several dB better than the performance without the limiter. For the polar codes, the non-statistical and statistical performances are very close, and use of the limiter makes no significant difference. We conjecture that this is because the SC decoding algorithm is sequential, whereas the SP and BCJR decoding algorithms are essentially iterative. In the case of the impulsive noise channel without the limiter, non-statistical decoding performance for the LDPC code is worse than statistical and both are poor. With the limiter, the LogAL-SP algorithm becomes better than LogSP algorithm, and both are much better overall, by about 12dB. For the turbo and polar codes, non-statistical and statistical performance is almost the same both with and without the limiter, but there is about a 7dB overall advantage when using the limiter.

The processing complexity (i.e., not including initialization and channel estimation) of the LogAL-SP decoding algorithm is about 8 % lower than that of the LogSP algorithm, over all three channels. The processing complexities of the LogAL-BCJR and LogBCJR algorithms are about the same, as are those of the LogAL-SC and LogSC algorithms, again over all channels. However, for both code classes, initialization of the algorithms is simpler in the non-statistical case. Of course, over all channels, the non-statistical algorithms have the advantage that estimation of the channel noise variance is not required. On the other hand, there is also a need to implement a quotient, which is preferably avoided in FPGA due to its implementation complexity. In the non-statistical case the algorithm can be initialized directly from the values of the channel samples y.

To conclude, this paper has shown that there is no loss of performance, nor increase in complexity, and sometimes a significant improvement in both, when using non-statistical AL decoding algorithms with the universal Euclidean metric over AWGN, fading and impulsive noise channels. We have also shown that the use of a simple signal amplitude limiter can significantly improve the performance of both statistical and non-statistical decoding algorithms over fading as well as impulsive channels.

ACKNOWLEDGEMENTS

We are very grateful for discussions with several colleagues, and for the helpful comments and suggestions of the reviewers, which have significantly improved several aspects of this paper. This research has been funded by Mar del Plata University and CONICET.

REFERENCES

[1] Gallager R. G. Low-density parity-check codes. *IRE Trans Info Theory*1962;(8):8–21.

[2] Berrou C., Glavieux A., Thitimajshima P. Near Shannon limit error-correcting coding and decoding: turbo codes. *Proc IEEE Int. Conf. on Communications* 1999;p. 1064–1070.

[3] Arikan E. Channel Polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Trans. Inf.Theory*2009;(55):3051–3073.

[4] Eroz M., Feng-Wen S., Lin-Nan L. DVB-S2 Low Density Parity Check Codes with Near Shannon Limit Performance. *International Journal of Satellite Communications and Networking*, Int J SatellCommun Network2004 2004;(22):269–279.

[5] Nurellari E., LDPC Coded OFDM and Its Application to DVB-T2, DVB-S2 and IEEE 802.16e; 2012.

[6] Neha A. K. Optimization of FEC In IEEE 802.16D. *International Journal of Enterprise Computing and Business Systems*2012;(2,1):1–12.

[7] Banerji S., Chowdhury RS. On IEEE 802.11: Wireless LAN Technology. *International Journal of Mobile Network Communications and Telematics (IJMNCT*) 2013;(3,4).

[8] Mankar M., Asutkar G., Dakhole P. Reduced Complexity Quasi-Cyclic LDPC Encoder for IEEE 802.11N. *International Journal of VLSI design and Communication Systems (VLSICS)*2016;(7,5/6):33–47.

[9] Jain S., Koshti R. Performance Improvement of IEEE 802.22 WRAN with Different FEC Codes. *International Conference on Machine Intelligence Research and Advancement 2*015;(2,7):244–250.

[10] Cohen A. E., Parhi K. K. A Low-Complexity Hybrid LDPC Code Encoder for IEEE 802.3an (10GBase-T) Ethernet.*IEEE Transactions on Signal Processing*2009;(57,10):4085–4094.

[11] Consultative Committee for Space Data Systems.https://public.ccsds.org/about/default.aspx.

[12] Douillard C., Jezequel M., Berrou C., Brengarth N., Tousch J, The turbo code standard for DVB-RCS;. Available at https: //www.academia.edu/.... • . • ./The_Turbo_Code_Standard_for_DVB-RCS (2009).

[13] Bioglio V., Condo C., Land L. Design of Polar Codes in 5G New Radio. Polar codes and Its Application in Speech Communication, (https://arxivorg/abs/180404389v1) 2018;p. 1–9.

[14] Zhao S., Sji P. A.,Wang B. Polar codes and Its Application in Speech Communication. *International Conference onWireless Communications and Signal Processing (WCSP)* 2011;p. 1–14.

[15] Forney G. D. Concatenated codes. The MIT Press; 1966.

[16] Tanner L. M. A recursive approach to low complexity codes. *IEEE Trans. Info. Theory* 1981;(5):533–547.

[17] Wiberg N., Loeliger H. A., Kotter L. Codes and Iterative Decoding on General Graphs. *Euro Trans.Telecommun.*1995;(6):516–526.

[18] Bahl L., Cocke J., Jelinek F., Raviv J. Optimal decoding of linear codes for minimising symbol error rate. *IEEE Trans. Info.Theory.*1974;(2):284–287.

[19] Pearl J. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Series in Representation and Reasoning; 1988.

[20] Saeedi H., Banihashemi A. Performance of Belief Propagation for Decoding LDPC Codes in the Presence of Channel Estimation Error. *IEEE Trans. on Comm.* 2007;(1):83–89.

[21] Moon T.K. Error Correction Coding. Wiley-Interscience; 2005.

[22] Valenti M.C., Sun J. The UMTS turbo code and an e_cient decoder implementation suitable for software-de_ned radios.*International journal of wireless information networks*2001;(8):203–215.

[23] Farrell P. G., Arnone L., CastiñeiraMoreira J. Euclidean distance soft-input soft-output decoding algorithm for LDPC codes. *IET Communications.*2011;(16):2364–2370.

[24] Arnone L., CastiñeiraMoreira J., Farrell P. G. Field programmable gate arrays implementations of lowcomplexity soft-input soft-output low-density parity-check decoders. *IET Communications* 2012;(12):1670–1675.

[25] Castiñeira Moreira J., Farrell P. G. Essential of Error-Control Coding. JohnWiley and sons; 2006.

[26] Wassinger N., Liberatori M. C., Castiñeira Moreira J. Variación de la performance de decodificadores LDPC de distanciaEuclideana con la base logarítmicautilizada. *Revista Argentina de Ingeniería, RADI*2012;(1):75–83.

[27] Reed M. C., Asenstorfer J. A. A novel variance estimator for turbocodedecoding.*ProcIntConf on Telecommunications*1997;(1):173–178.

[28] MacKay DJC, Neal RM. Near Shannon limit performance of low density parity check codes. *IET Electronics Letters.*1997;(6):457–458.

[29] Mackay D., Gallager codes, MN codes and review papers on sparse graph codes.Available at http://www.inference phy.cam.ac.uk/mackay/CodesGallager.html.

[30] Hanzo L. T. H. L., Yeap B. L. Turbo Coding, Turbo Equalisation and Space-Time Coding for Transmission overWireless Channels. JohnWiley and sons; 2002.

[31] Worm A., Hoeher P., Wehn N. Turbo Decoding Without SNR Estimation. *IEEE Communications Letters* 2000;(6):193– 195.

[32] Leroux C., Raymod A., Sarkis A., Gross W. A Semi-Parallel Successive-Cancellation decoder for polar codes.*IEEE Trans. on Sig. Proc.* 2013;(3):289–299.

[33] Liberatori M. C., Arnone L., Castiñeira Moreira J., Farrell P. G. Soft-Distance Metric Decoding of polar codes. *Proc. Int. Conf. on Cryptography and Coding* (IMACC 2015) 2015;pp. 173–183.

[34] Hou J., Siegel P. H.,Milstein L. B. Performance analysis and code optimization of low density parity check codes on Rayleigh fading channels. *IEEE Journal on Selected Areas in Communications*2001;(15):924–934.

[35] Tse D., Viswanath P. FundamentalsofWireless Communication.Cambridge University Press; 2005.

[36] Johnston M., Sharif B. S., Tsimenidis C, Chen L. Sum-Product Algorithm Utilizing Soft Distances on Additive Impulsive Noise Channels. *IEEE Trans. on Comm.*2013;(6):2113–2115

.

[37] Arnone L., Liberatori M., Petruzzi D., Castiñeira Moreira J. Performance of a Simplified Soft-Distance decoding algorithm for LDPC codes over the Rayleigh fading channel. *Latin American Applied Research.*2013;(43):219–223.

[38] Middleton D. Statistical-physical models of electromagnetic interference. *IEEE Trans. on Electromagnetic Compatibility.* 1977;(3):106–127.

AUTHORS

Mónica C. Liberatori was born in Entre Ríos, Argentina, in 1961. She obtained her MSc in Network Data in La Plata University, Argentina, in 2006. She received her Electronics Engineer degree from Mar del Plata University, Argentina, in 1993. She is currently a Professor with the Engineering School, Mar del Plata University on Network data, Digital Communications, Encryption and Error-Control Coding. His research interests include the broad area of digital communications systems, error-control coding, data and Wireless networks.

Leonardo J. Arnone was born in Mar del Plata, Argentina, in 1964. He received his PhD in Electronics Engineering from Mar del Plata University, in 2008. He received a MSc degree in Informatics from Autonomous Barcelona University, in 1997. He is currently an Associate Professor with the Engineering School, Mar del Plata University on electronics devices. His research interests are on programable electronics and implementation issues of digital communications techniques, error-control coding implementations.

Jorge Castiñeira Moreira was born in Buenos Aires, in 1962. He received his Electronics Engineering degree from Mar del Plata University, in 1990. He received his MSC degree and PhD degree form Lancaster University, Lancaster, UK, in 1996 and 2000, respectively. He is currently a Professor with the Engineering School, Mar del Plata University, on Digital Communications, Encryption and Error-Control Coding. He is also an independent Researcher with CONICET. His research interests include the broad area of digital communications systems, information theory, error-control coding, data and Wireless networks. He is co-author of the book "Essentials of Error-Control coding", Wiley, 2006.

Patrick Guy Farrell was born in Buenos Aires, Argentina, in 1938. He received the BSc in Electrical Engineering from City University, London, UK, in 1965. He did research on coding for noisy data links at the University of Cambridge, and was awarded the PhD in 1969. His research, teaching and consultancy interests include error-correcting codes, information theory, digital communications, mobile radio, signal processing and electronic circuits, and he is author or co-author of over 300 papers, reports and presentations, including the book "Essentials of Error-Control Coding, Wiley, 2006". He received the Marconi Premium of the IET. He is a Fellow of the Royal Academy of Engineering and the IET, a Life Fellow and Millenium Medalist of the IEEE, and a Fellow of the IMA.