A New Iterative Threshold Decoding Algorithm for One Step Majority Logic Decodable Block Codes

M. Lahmer
High School of Technologies
Km5, Route gouray *B.P*:3103
Toulal Meknes - Morroco

M. Belkasmi
National Institute of Computer Sciences and Systems Analysis (ENSIAS)
B.P 713 – *ENSIAS* AGDAL
RABAT-Morroco

ABSTRACT
The performance of iterative decoding algorithm for one-step majority logic decodable (OSMLD) codes is investigated. We introduce a new soft-in soft-out of APP threshold algorithm which is able to decode theses codes nearly as well as belief propagation (BP) algorithm. However the computation time of the proposed algorithm is very low. The developed algorithm can also be applied to product codes and parallel concatenated codes based on block codes. Numerical results on both AWGN and Rayleigh channels are provided. The performance of iterative decoding of parallel concatenated code (17633,8595) with rate 0.5 is only 1.8 dB away from the Shannon capacity limit at a BER of 10^{-5}.

General Terms
Information theory and Coding, Signal Processing.

Keywords
One step majority logic decodable codes, OSMLD, iterative threshold decoding, parallel concatenated block codes, product codes.

1. INTRODUCTION
TURBO codes or LDPC codes are currently the most effective solution for Forward Error-Correction (FEC) in applications requiring either high code rates (R > 0.8), very low error floors, or low-complexity decoders able to operate at several hundreds of megabits per second and even higher. Practical implications of theses codes are numerous namely 3G wireless phones, Digital Video Broadcasting (DVB) systems, or Wireless Metropolitan Area Networks (WMAN). Turbo decoding relies on the exchange of probabilistic messages (the so-called *extrinsic* information) between two *soft-input soft-output* (SISO) decoders for the component convolutional codes. However, for concatenated schemes with block component codes, the computational complexity of trellis-based SISO decoding algorithms is often high. This has led to look for new SISO decoding algorithms with low complexity and high performance, e.g., [2]-[4]. These algorithms calculate extrinsic information using classical decoders such as Chase algorithm in [2], ordered statistics decoding algorithm in [4] and Hartmann/Rudolph algorithm in [3]. In this perspective we present a new iterative decoding algorithm based on a SISO extension form of threshold algorithm [8]. The proposed algorithm is attractive for three reasons: (1) It can be simply implemented; (2) the decoding delay is short; and (3) it has good performances. The use of the threshold algorithm in iterative decoding was introduced for the first time by Svirid [7] but for convolutional codes. Our iterative decoding process follows that given by Pyndiah in [2]. However, instead of using an extension of Chase algorithm on BCH codes, we will apply an extension of Massey algorithm on one step majority logic decodable (OSMLD) codes. On the other hand Lucas et al. [3] introduce an iterative decoding algorithm for several families of codes (e.g. OSMLD codes) but they use an approximation of Hartmann/Rudolph algorithm. This paper is focused on iterative decoding of OSMLD codes.

The organization of the paper is as follows. In Section 2, the basic concept of SISO Threshold decoding algorithm is introduced. In Section 3, we describe iterative decoding algorithm. Section 4 is dedicated to simulation results of BER performance for different parallel concatenated codes based on OSMLD codes.

2. SOFT-IN/SOFT-OUT THRESHOLD DECODINGAGE SIZE
2.1 One Step Majority Logic Codes
Consider an (n, k) linear code C with parity-check matrix H. The row space of H is an (n, n-k) cyclic code, denoted by $C^{\perp}$, which is the dual code of C, or the null space of C. For any vector v in C and any vector w in $C^{\perp}$, the inner product of v and w is zero.

Now suppose that a code vector in C is transmitted over a binary symmetric channel. Let e (e_1, e_2, …., e_n) and R (r_1, r_2, …, r_n) be the error vector and the received binary vector respectively. Then R = v + e. For any vector w in the dual code $C^{\perp}$, we can form the following linear sum of the received digits:

$$A = \sum_{p=1}^{n} r_p w_p$$

Which is called a parity-check sum. Using the fact that <w,v>=0, we obtain the following relationship between the check sum A and error digits in **e**:

$$A = \sum_{p=1}^{n} e_p w_p$$

Suppose that there exist J vectors in the dual code $C^{\perp}$, which have the following properties:

1. The j^{th} component of each vector w_i is a 1
2. For $i \neq j$ there is at most one vector whose i^{th} component is a 1

These J vectors are said to be orthogonal on the j^{th} digit position. We call them orthogonal vectors. Now, let use form J parity-check sums from these J orthogonal vectors,

For each i in {1,.., J} $A_i = \sum_{p \neq j} e_p + e_j$

we see that the error digit e_j is checked by all the check sums above. Because of the second property of the orthogonal vectors, any error digit other than e_j is checked by at most one check sum. These J check sums are said to be orthogonal on the error digit e_j.

If all the error digits in the sum A_i are zero for $i \neq j$, the value of A_i is equal to e_j. Based on this fact, the parity-check sums orthogonal on e_j can be used to estimate e_j, or to decode the received digit r_j.

Table 1 shows some examples of OSMLD codes that can be decoded with the proposed SISO decoder described above. In this table we used the abbreviations DSC for Difference Set Cyclic codes, EG for Euclidean Geometry codes and BCH for Bose Chaudhuri and Hocquenghem codes. The EG codes used in this study are 0-order and, for an extensive description of projective geometry codes and Euclidean geometry can be found in [10].

Table 1. Set of OSMLD Codes

N	***k***	***J***	d_{min}	***Rate***	***code***
7	3	3	4	0.42	DSC
15	7	4	5	0.46	BCH
21	11	5	6	0.52	DSC
63	37	8	9	0.58	EG
73	45	9	10	0.61	DSC
255	175	16	17	0.68	EG
273	191	17	18	0.69	DSC
819	447	15	16	0.54	OSMLD
1057	813	33	34	0.76	DSC
4161	3431	65	66	0.82	DSC
16513	14325	129	130	0.86	DSC

2.2 Majority Logic Decoding Principle

The error digit ej is decoded as 1 if at least one-half of the check sums orthogonal on ej, are equal to 1; otherwise, ej is decoded as 0 like majority rule. When C is a cyclic code, each ei can be decoded simply by cyclically permuting the received word r into the buffer store.

Example: let us consider the (7,3) code, which is the short code in difference set codes class (see Table 1). This code is specified by the perfect difference set P={0, 2, 3} of order 2. From this prefect set, we can form the following three check sums orthogonal on e_7:

$$\begin{cases} A_1 = e_4 + e_5 + e_7 \\ A_2 = e_2 + e_6 + e_7 \\ A_3 = e_1 + e_3 + e_7 \end{cases}$$

The If a simple error e=(000001) occur, than we have $A_1 = A_2 = A_3 = 1$. If a double error occur, as an example $e_7=1$ and one value of $e_1, ..., e_6$ is equal to 1, then two values of Ai are 1. So we can say that :

- $e_7=1$ if only and if at least 2 values of Ai are 1
- $e_7=0$, otherwise.

2.3 Soft-input Soft-output Threshold Decoding

Threshold decoding is simply the logical extension to soft decisions of majority decoding. Soft-out decoding algorithm when applied to OSMLD codes is stated as follows:

Let us consider a transmission of block coded binary symbols {0,1} using a BPSK modulation over AWGN channel, the decoder soft output for the j^{th} bit position of a given soft input $\mathbf{R}(r_1, r_2, .., r_n)$ is defined as:

$$LLR_j = \ln\left[\frac{P(c_j = 1/R)}{P(c_j = 0/R)}\right] \quad (1)$$

where $C(c_1, c_2, ..., c_n)$ is the transmitted codeword. The hard decision vector corresponding to the received vector r is denoted by $H(h_1, h_2, ..., h_n)$.

For a code with J orthogonal parity check equations, (1) can be expressed as:

$$LLR_j \approx \ln\left[\frac{P(c_j = 1/\ B_i}{P(c_j = 0/\ B_i}\right] \quad (2)$$

where B_i, i=1...J are obtained from the orthogonal parity check equations on the j^{th} bit as follows:

The term B_0 is defined to be $B_0 = h_j$. Each of the $\mathbf{B_i}$ i=1,..,J is computed by dropping the term h_j from the i^{th} orthogonal parity equation. Thus, each of the B_i can be written as:

$$B_i = c_j + \sum_{k=1}^{n_i} e_{ik} \quad (3)$$

Where e_{ik} is the k^{th} error term in the i^{th} parity check equation excluding e_j (n_i is the total number of terms in the i^{th} orthogonal parity equation without c_j).

By applying BAYES rule, (2) becomes

$$LLR_j \approx \ln\left[\frac{P(\ B_i\ / c_j = 1) \times P(c_j = 1)}{P(\ B_i\ / c_j = 0) \times P(c_j = 0)}\right] \quad (4)$$

Since the parity check equations are orthogonal on the j^{th} symbol the individual probabilities $P(B_i / c_j = 1 \text{ or } 0)$ are all independent and (4) can be rewritten as

$$LLR_j \approx \sum_{i=0}^{J} \ln\left[\frac{P(B_i / c_j = 1)}{P(B_i / c_j = 0)}\right] + \ln\left[\frac{P(c_j = 1)}{P(c_j = 0)}\right] \quad (5)$$

Assume that the transmitted symbols are equally likely to be 0 or 1, and thus the last term in (5) is null. As a result, we obtain

$$LLR_j \approx \sum_{i=1}^{J} \ln\left[\frac{P(B_i / c_j = 1)}{P(B_i / c_j = 0)}\right] + \ln\left[\frac{P(B_0 / c_j = 1)}{P(B_0 / c_j = 0)}\right] \quad (6)$$

According to [9], (6) can be expressed as

$$LLR_j \approx (1-2B_0)w_0 + \sum_{i=1}^{J}(1-2B_i)w_i \quad (7)$$

where the value of $(1-2B_i)$ is equal to +1 or –1 and w_i is a weighting term proportional to the reliability of the i^{th} orthogonal parity check. It is easy to show that:

$$(1-2B_o)w_o = \frac{4E_s}{N_o} r_j \quad (8)$$

and

$$w_i = \ln\left[\frac{1+\prod_{k=1}^{n_i} \tanh(L_{ik}/2)}{1-\prod_{k=1}^{n_i} \tanh(L_{ik}/2)}\right] \quad (9)$$

where *ik* represents the k^{th} element of the i^{th} parity equation and

$$L_{ik} = \frac{4E_s}{N_0}|r_{ik}|. \quad (10)$$

Thus the soft output can be split into two terms, namely into a normalized version of the soft input r_j and an extrinsic information E_j representing the estimates made by the orthogonal bits on the current bit c_j. Hence (7) becomes

$$LLR_j = \frac{4E_s}{N_0} r_j + E_j \quad (11)$$

We make the following notations:

$$L_c = \frac{4E_s}{N_0}, \quad (12)$$

which is called the reliability value of the channel.

The algorithmic structure of the SISO threshold decoding can be summarized as follows:

For each $j = 1,..,n$

Compute the terms B_i *and* w_i, $i \in 1,..,J$

Calculate B_0 *and* w_0

Compute the extrinsic information E_j

The Soft-output is obtained by: $LLR_j = L_c r_j + E_j$

2.4 Modifications for Rayleigh fading channel

For our algorithm to be applicable in wireless environment, their performance on fading channels must be examined. In the channel model we use, each received bit r_j can be expressed as:

$$\mathbf{r_j} = \mathbf{a_j}\hat{\mathbf{c}}_\mathbf{j} + \mathbf{n_j} \quad (13)$$

In this representation, $\hat{c}_j$ is a BPSK symbol associated to the transmitted bit c_j, and n_j is an AWGN. The Rayleigh variable a_j is generated as

$$\mathbf{a_j} = \sqrt{\mathbf{x_j^2} + \mathbf{y_j^2}} \quad (14)$$

where x_j and y_j are zero mean statistically independent Gaussian random variables each having a variance σ^2. We consider the power normalized to one as

$$\mathbf{E}\left[\mathbf{a}_\mathbf{j}^2\right] = 2\sigma^2 = 1 \quad (15)$$

which give a variance of 0.5 for Gaussien variables.

On the Rayleigh fading channel, the availability of channel side information is the key issue in determining the necessary modification for the iterative threshold decoding algorithm. The threshold decoding algorithm has to be modified slightly by changing equation (12) which define the reliability value of the channel by

$$\mathbf{L_c} = \frac{4\mathbf{E_s}}{\mathbf{N_0}}\mathbf{a_j} \quad (16)$$

3. ITERATIVE THRESHOLD DECODING

3.1 Iterative Threshold Decoding of OSMLD codes

Iterative decoding process (see Fig. 1) can be described as follows: In the first iteration, the decoder only uses the channel output as input, and generates extrinsic information for each symbol. In subsequent iterations, a combination of extrinsic information and channel output is used as input

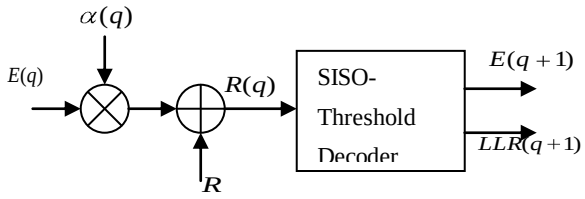

Fig. 1: The block diagram of the q^{th} iteration.

The soft input respectively the soft output of the q^{th} iteration is given by:

$$\mathbf{R(q)} = \mathbf{R} + \alpha\mathbf{(q)E(q)} \quad (17)$$

$$\mathbf{LLR(q)} = \mathbf{L_c R(q)} + \mathbf{E(q+1)} \quad (18)$$

where R represent the received data, E(q) is the extrinsic information computed by the previous iteration. In our procedure we use a fixed value 1/J for the parameter $\alpha(q)$ and this for all iterations. The value chosen for $\alpha(q)$ reacts as an average of all J estimators which contribute in the computation of E_j.

3.2 Iterative Threshold Decoding of product codes

The developed algorithm can also be applied to product codes and parallel concatenated codes based on block codes. Let us consider two linear block codes C^1 having parameters (n_1, k_1, d_1) and C^2 having parameters (n_2, k_2, d_2) where n_i, k_i and d_i (i=1,2) stand for codeword length, number of information bits and minimum Hamming distance respectively. It is assumed that the information symbols are the first k_1 symbols of C^1 and the first k_2 symbols of C^2. The product code $PC = C^1 \otimes C^2$ is an $(n_1 n_2, k_1 k_2, d_1 d_2)$ code whose codeword's are constructed by encoding $k_1 \times k_2$ information symbols with code C^1 and the resulting $\mathbf{k}_2 \times \mathbf{n}_1$ symbols with C^2 (see **Fig. 2**).

A parallel concatenated block (PCB) code can be constructed by a parallel concatenation of block codes. The PCB code is a PC but without the checks on checks part (see **Fig. 2**). The rate of product code PC and PCB codes are given respectively by

$$R_{PC} = \frac{k_1 \times k_2}{n_1 \times n_2} \text{ and}$$

$$R_{PCB} = \frac{k_1 \times k_2}{(n_1 \times n_2) - [(n_1 - k_1) \times (n_2 - k_2)}.$$

The major disadvantage of the parallel concatenated code is the loss in minimum distance. It is only $(\mathbf{d}_1 + \mathbf{d}_2) - 1$ compared to $\mathbf{d}_1 \cdot \mathbf{d}_2 - 1$ for the product code.

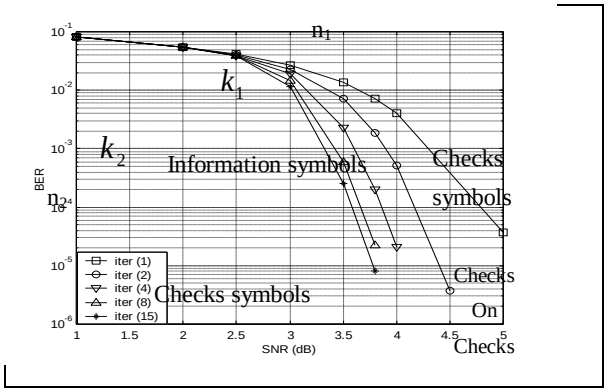

Fig. 2 : Construction of a Product code

The decoding procedure of product code is performed by cascading elementary decoders (rows and columns). On receiving matrix [R], the first decoder performs the soft decoding of the rows (or columns) using as input matrix [R]. Soft Input / Soft Output decoding is performed using the new proposed algorithm.

Tables 2 show some examples of constructed product and PCB codes by using codes in table 1.

Table 2 Set of PC and PCB codes based on OSMLD Codes

Constructed code	*Component code (C^1)*	*Component code (C^2)*	*Rate*
PCB (161,49)	BCH (15,7)	BCH (15,7)	0.30
PCB (341,121)	DSC (21,11)	DSC (21,11)	0.35
PCB (1295,495)	DSC (21,11)	DSC (73,45)	0.38
PCB (3293, 1369)	EG(63,37)	EG(63,37)	0,41
PCB (4545,2025)	DSC (73,45)	DSC (73,45)	0.45
PCB (17633,8595)	DSC (73,45)	DSC (273,191)	0.48
PCB (67805, 36481)	DSC(273,191)	DSC(273,191)	0.53
PC (1533,495)	DSC(21,11)	DSC(73,45)	0.32
PC (4095,1337)	BCH (15,7)	DSC(273,191)	0.32
PC (3969,1369)	EG (63,37)	EG (63,37)	0,34
PC (5329,2025)	DSC(73,45)	DSC(73,45)	0.37
PC (19929, 8595)	DSC(73,45)	DSC(273,191)	0.43

4. SIMULATION RESULTS

This section considers simulation results and analysis for some OSMLD codes, product codes and parallel concatenated blocks codes. We would like to notify that for all our simulations we have used a minimal residual error bit of 200 and residual error block of 30. We have used a number of iterations such that there is no significantly more gain by more iteration. The performance improves with each iteration in all simulation results presented.

4.1 Performances of OSMLD codes

Figure 3 depicts the performance of iterative decoding of (1057, 813) DSC code with rate 0.76. We can see that the performance improves with each iteration. The first iteration show the performances of classical threshold decoding [8].

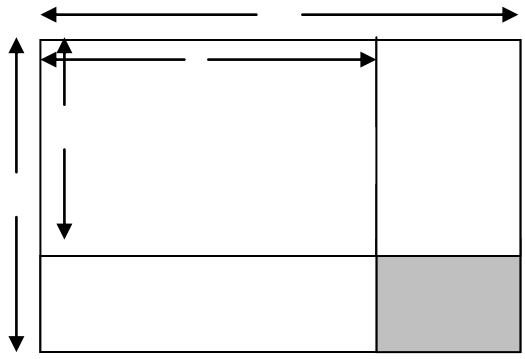

Fig.3 : BER performance of (1057, 813) code on AWGN channel

Figure 4 shows the frame error rate (FER) performance of (73,45), (273,191), (819,447) and (1057,813) codes on both AWGN and Rayleigh fading channel. As it can be seen the slope of the frame-error rate (FER) curve is as steep as for the Gaussian channel. It is worth mentioning that the number of

iterations needed for Rayleigh fading channel is about the same as for the AWGN channel.

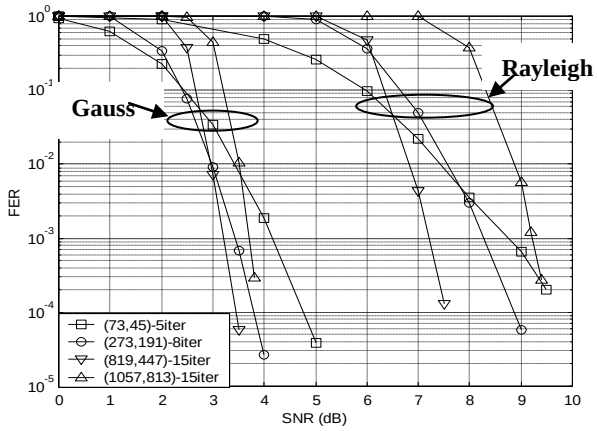

Fig. 4**:** FER performance of (73,45), (273, 191), (819, 447) and (1057,813) codes on AWGN and Rayleigh fading channels

OSMLD codes are LDPC(Low Density Parity Check) codes and can be decoded by belief propagation (BP) algorithm [11]. BER Performance of iterative decoding for (73,45) and (273,191) codes are shown and compared to those of BP[11] in Figure 5. As it can be seen our results are worse by 0.2 dB at BER 10^{-5}. This coding gain is negligible compared to the required complexity (see Table 3).

Table 3 shows simulation time to decode 1000 frame of (73, 45), (273,191), (819,447) and (1057, 813) codes with our algorithm and BP algorithm using 15 iterations (by using a computer with Pentium 4, 3.06 GHz). We can observe that as the code length increases the computational time complexity of BP increases compared to that of our iterative decoding algorithm.

4.3 Performances of product codes

This section considers simulation results and analysis for some PC and PCB codes, all of which use one step majority logic decodable (OSMLD) component codes (see Table 2).

In Fig.6, we present the simulation results for the (4545, 2025) PCB code of rate 0.45 constructed from (73, 45) DSC code. We can see that the improvement is great for the first iterations and is negligible after the 16^{th} iteration. Here we recognize the Turbo effect.

In Fig.7, we present a comparison between our simulation results and results published in [3] for the (4545, 2025) PCB code. Although they use a modified Gaussian channel with a tanh function modulation, their results are worse by 0.6 dB at a BER of 10^{-4}.

The Fig. 8 shows that performance results after 50 iterations for the (17633,8595) PCB code is only 1.8 dB away from the Shannon capacity limit at a BER of 10^{-5}.

In Fig.9, we present the simulation results for the (1533,495) asymmetric PC code of rate 0.32 constructed from (21, 11) and (73, 45) DSC code. As can be seen, performance increases with each iteration.

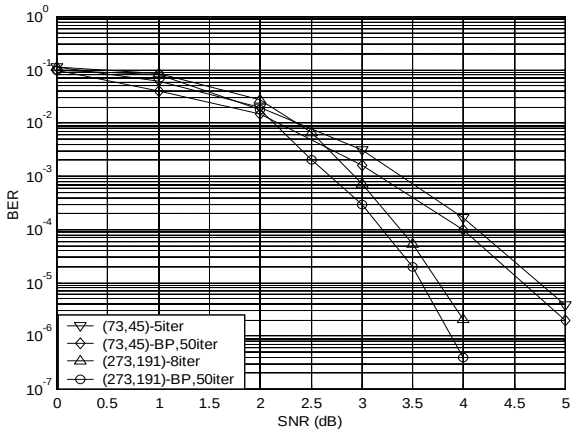

Fig. 5: Performance of (73,45) and (273,191) in comparison with BP on AWGN channel

Table 3 : Comparison in terms of time computation and error performances

Codes	Our algorithm	BP algorithm	Gain of BP at BER=10^{-5}
(73,45)	4[s]	23[s]	0.2 dB
(273,191)	58[s]	300[s]	0.2 dB
(1057,813)	801[s]	13000[s]	0.45 dB

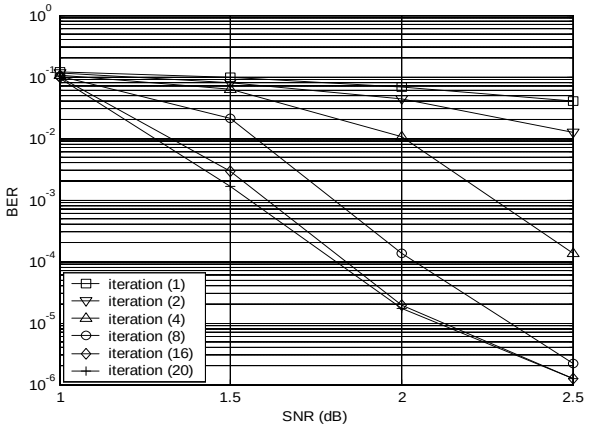

Fig. 6 Performance of iterative decoding of the (4545,2025) code on AWGN channel

The curves in Fig.10 present the simulation results for the (1253,495), (4545,2025) PCB codes and (5329,2025) PC. The PC is compared to two PCB codes. In the first case, the PC is compared to the PCB code constructed from the same component codes ((73, 45) DSC code). As envisaged, the PC outperforms the PCB code. In the second case, the PC is compared to a PCB code with same rate. As can be seen, the PC is better than PCB code. It seems that contrary to the PCB codes, product codes doesn't have error floor. Furthermore, product codes are better than PCB codes for larger SNR.

In Fig.11, we present a comparison between our simulation results for (1533, 495) product code and results published in [12] for convolutional turbo code of rate 1/3 using 16 states. We can observe that product code have worse performance at low SNR compared to convolutional code, whereas at SNR> 4.3 dB, the product code are better.

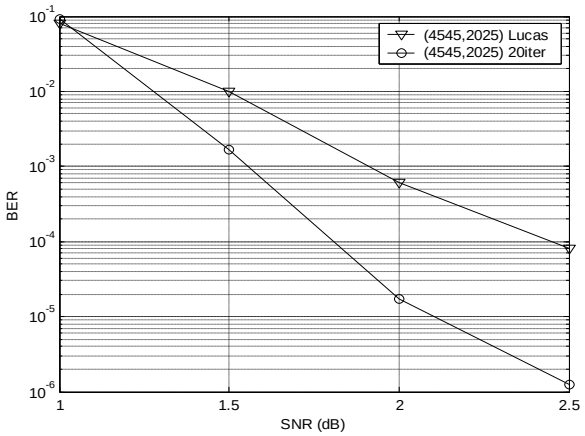

Fig. 7 **:** Performance of iterative decoding of the PCB(4545, 2025). The BER of the Lucas et al.[3] algorithm is given as reference.

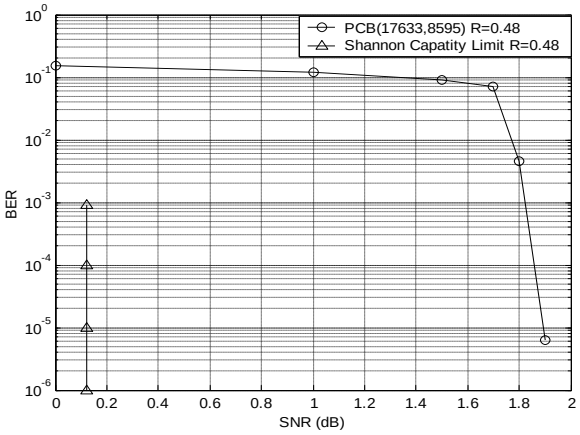

Fig. 8 : Performance of the (17633,8595) PCB code compared to Shannon capacity limit

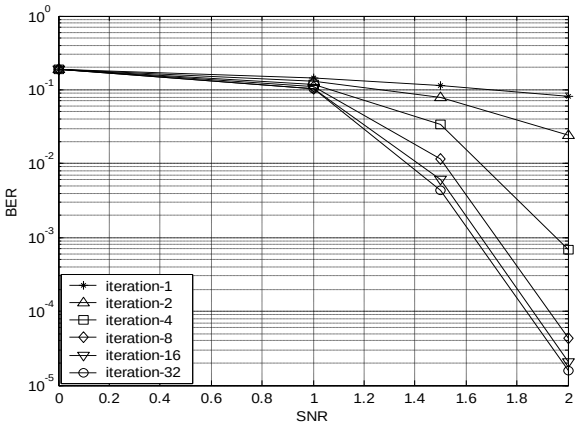

Fig. 9 : Performance of iterative decoding of the (1533, 495) product code on AWGN channel

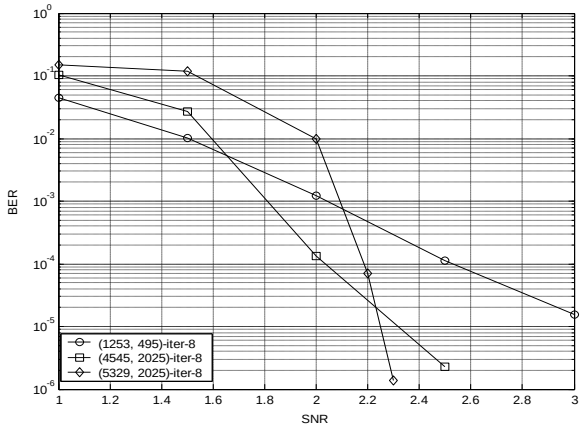

Fig. 10 **:** Performance of iterative decoding of the PC(5329,2025),PCB(4545,2025) and PCB(1253,495) codes

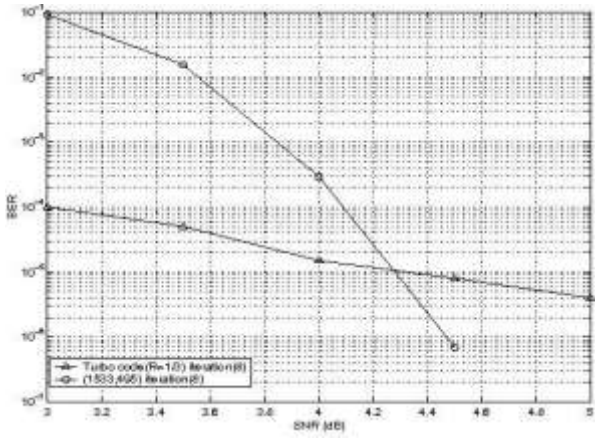

Fig. 11: Performance of iterative decoding of the (1533, 495) product code on Rayleigh fading channel

5. CONCLUSION

In this paper we have presented a new iterative threshold decoding algorithm for simple codes, product codes and parallel concatenated block codes based on one-step majority logic decodable codes. We use an extension of Massey's algorithm [8] as a Soft input/ Soft-output component decoder. The structure of our iterative decoder follows the model of Pyndiah [2] with some modifications. This algorithm has been tested on several codes based on OSMLD codes and good performances have been obtained over the Gaussian and Rayleigh fading channels. It is interesting to extend this iterative decoding algorithm on quasi-cyclic and multi-step majority logic decodable codes.

6. REFERENCES

[1] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding : Turbo-codes (1)," IEEE Int. Conf. on Comm. ICC'93, Geneva, May 1993, pp. 1064-1071.

[2] R. Pyndiah, "Near-Optimum Decoding of Product Codes: Block Turbo Codes," IEEE Trans. Commun., Aug. 1998, Vol. 46, N° 8, pp. 1003-1010.

[3] R. Lucas, M. Bossert and M. Breitbach, "On Iterative Soft-Decision Decoding of Linear Binary Block Codes and Product Codes, " IEEE Journal on selected areas in communications, February 1998, Vol.

[4] M. P. C. Fossorier and *S.* Lin. "Soft-Input Soft-Output Decoding of Linear Block Codes Based on Ordered Statistics," Proc. 1998 IEEE Global Telecomm. Conf. (GLOBECOM'98), Sydney, Australia Nov. 1998. pp. 2828-2833,

[5] J. Hagenauer, E. Offer, and L. Papke, "iterative decoding of binary block and convolutional codes," IEEE Trans. Inform. Theory, Mar. 1996, Vol. 42, pp. 429-446.

[6] R. Lucas, M. P. C. Fossorier, Yu Kou, and Shu Lin, "Iterative Decoding of One-Step Majority Logic Decodable Codes Based on Belief Propagation," IEEE Trans. Commun, June 2000, VOL. 48, NO. 6, pp.931-937.

[7] Yuri V. Svirid and Sven Riedel, "Threshold Decoding of Turbo-Codes," IEEE Int. Symposium on Information Theory, 1995, pp. 39.

[8] J.L Massey, "Threshold Decoding," Cambridge, Ma, M.I.T. Press, 1963.

[9] C. Clark and B. Cain, "Error-Correction Coding for digital communications," Plenum Press, 1981.

[10] S. Lin and D. J. Costello, "Error Control Coding, Fundamentals and Applications," Englewood Cliffs, NJ: Prentice-Hall, 1983.

[11] R. Lucas, M. P. C. Fossorier, Yu Kou, and Shu Lin, "Iterative Decoding of One-Step Majority Logic Decodable Codes Based on Belief Propagation," IEEE Trans. Commun, June 2000, VOL. 48, NO. 6, pp.931-937

[12] S. A. Barbulescu, "Iterative decoding of turbo codes and other concatenated codes," thesis, University of South Australia, Februry 1996, pp 869-870.