Journal of Basic and Applied Engineering Research
Print ISSN: 2350-0077; Online ISSN: 2350-0255; Volume 1, Number 11; October-December 2014 pp. 1-6

http://www.krishisanskriti.org/jbaer.html

Performance Analysis of Turbo Codes under Different Parameters in AWGN Environment

Mudita Chandra[1] and Sanjay Kumar Soni[2]

[1]M.Tech Student Dept. of Electronics & Communication Engineering Delhi Technological University Delhi-110042, India
[2]Dept. of Electronics & Communication Engineering Delhi Technological University Delhi-110042, India
E-mail: 1cmudita1989@gmail.com, 2sanjoo.ksoni@gmail.comAbstract

Abstract: *Turbo Codes belong to the class of convolutional error correcting codes used in wireless communication applications like mobile communications, deep-sea communications, satellite communications and other fields. Turbo decoding is an iterative principle where the constituent decoders pass extrinsic information to each other with an interleaver or deinterleaver in between. The performance of turbo codes approaches the Shannon capacity limit making them most sought after. The present work through simulations analyses Turbo codes performance under different parameters and different situations like number of iterations, different encoder structures, interleaver types and puncturing. Simulation results show that Turbo codes givea considerably remarkable performance under these different situations and that the probability of error decreases with each successive iteration of Turbo decoding.The simulations were done on MATLAB with BPSK as the modulation scheme and channel being AWGN.*

1. INTRODUCTION

TheShannon's channel coding theorem draws an important conclusion that a noisy channel can be converted into an error free channel by using suitable coding technique [9]. Traditional modulation techniques don't achieve this rate of transmission but however combining it with some suitable error control coding, performance close to the channel capacity can be achieved. This is what that has been realized through Turbo Codes.

Turbo principle is actually a concept that can be applied whereever there is a concatenation of two trellises or two structured entities. The name Turbo comes from the reason of its iterative cyclic feedback mechanism employed for decoding similar to the turbo machines. These individual cycles which produce a small change add up to a big improvement in the turbo principle idea.

Turbo decoders employ serial or parallel concatenated soft decision based Maximum A Posteriori probability (MAP) decoders [4] which give improved performance than the earlier hard decision based viterbi decoders. The key to turbo decoder operation is the "message passing" between them. The message here is the extrinsic information which is basically the improved version of the apriori probability. The decoding operation is done by turbo decoder for several iterations and the probability of error decreases with each successive iteration. With higher number of iterations Turbo Codes subsequently achieve performance close to the Shannon's theorem. With such a remarkable performance even at low SNR's, turbo codes[6] have been used in a variety of applications like3G mobile communications, deep-sea communications, satellite communications and other power constrained fields.

In this paper performance of turbo codes close to Shannon limit is shown by simulations. Also the performance of turbo code has been analyzed for the encoder structures, puncturing, and number of iterations.

2. SYSTEM MODEL

The transmitter side consists of Turbo encoder and modulator. Whenpasses through communication channel errors due to fading and noise creep in. At receiver, first demodulation is done followed by turbo decoding.

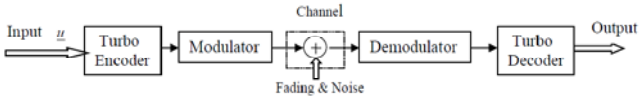

Fig. 1: Communication system model using Turbo decoding

2.1 Turbo Encoder

A basic turbo encoder is an encoder employs two recursive systematic convolutional encoders (may be identical or

different) in parallel [1, 3, 4] as shown in Fig 2. The input is passed through an interleaver which scrambles the input message and this interleaved message is fed to the second encoder. On the other hand original non interleaved version of the input message is simultaneously fed to the first encoder.

The use of two recursive convolutional encoders [4] in conjunction with the interleaver produces a code that contains very few code words of low weight having relatively few nearest neighbors. That is the code words are relatively sparse. Hence the coding gain achieved by a turbo code is due in part to this feature.

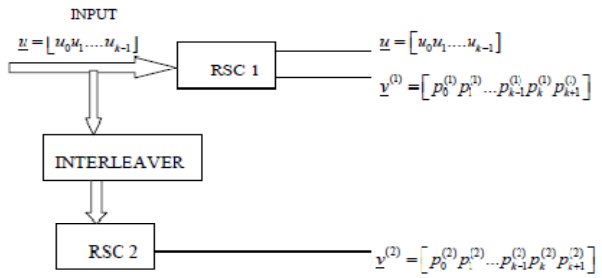

Fig. 2: Turbo Encoder

The recursive systematic convolutional encoder is like an IIR filter as shown in the Fig 3.

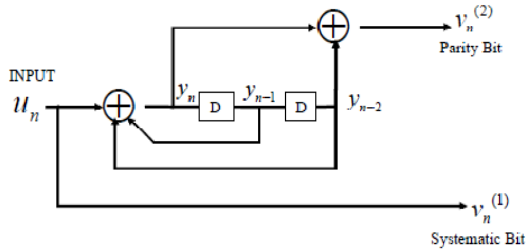

Fig. 3: Recursive Systematic Convolutional Encoder

Due to this IIR filter like structure it may be possible that an input sequencemay not yield a minimum weight codeword out of the encoder [4].The encoder output weight is thus kept finite by trellis termination. Through this process the encoder returns to the all zero state.

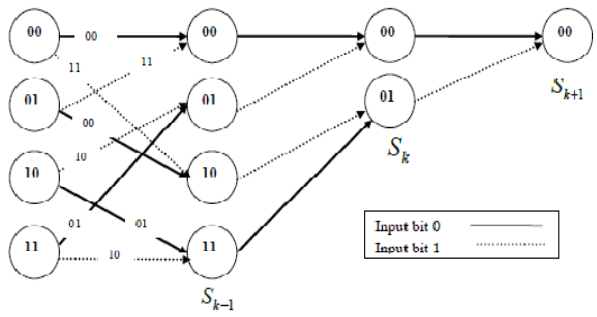

Fig. 4: Tailing Biting

This is done by tail biting in which the input is padded [7] with few bits so that the encoder may return to the all zero state. The usual practice is to zero terminate only the first encoder and not to zero terminate the second encoder. The trellis diagram with tail biting is shown in Fig 4.

2.2 Interleaver

In turbo codes the use of interleaver[3] accomplishes mainly two functions. Firstly, it spreads out the burst errors so that code words appear independent. Thus the burst error channel is transformed into random error channel at the input of the decoder and the code designed for independent error channels could be used for burst error channels. Secondly, the interleaver scrambles the data in such a way that the inputs to the two decoders are uncorrelated. So after corrections of some errors in the first component decoder, some of the remaining errors are spread by the interleaver in such a way that they get corrected by the second decoder.

2.3 Decoding Structure of Turbo Code

With input u the output code word of the turbo encoder of fig 3 is

$$\underline{v} = \left[u_0 p_0^{(1)} p_0^{(2)}, u_1 p_1^{(1)} p_1^{(2)} \ldots\ldots u_{k-1} p_{k-1}^{(1)} p_{k-1}^{(2)} \ldots \right].$$

After BPSK modulation this can be represented as

$$\underline{y} = \left[v_0 v_0^{(1)} v_0^{(2)}, v_1 v_1^{(1)} v_1^{(2)} \ldots\ldots v_{k-1} v_{k-1}^{(1)} v_{k-1}^{(2)} \ldots \right].$$

With the addition of noise in the communication channel the finally received signal is

$$\underline{r} = \left[r_0 r_0^{(1)} r_0^{(2)}, r_1 r_1^{(1)} r_1^{(2)} \ldots\ldots r_{k-1} r_{k-1}^{(1)} r_{k-1}^{(2)} \ldots \right] \text{ Or}$$

$$\left[\underbrace{r_0 r_1 \ldots r_{k-1} \ldots\ldots}_{r^{(s)}}, \underbrace{r_0^{(1)} r_1^{(1)} \ldots r_{k-1}^{(1)} \ldots\ldots}_{r^{(1)}}, \underbrace{r_0^{(2)} r_1^{(2)} \ldots r_{k-1}^{(2)} \ldots\ldots}_{r^{(2)}} \right]$$

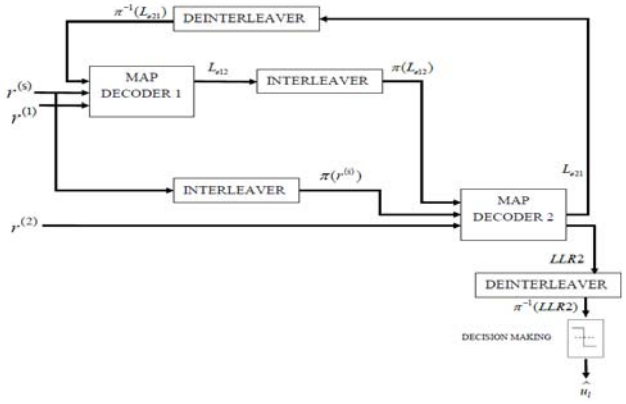

Fig. 5: Turbo Decoder

The turbo decoder [5] consists of two component decoders serially concatenated via an interleaver identical to the one used in the encoder [3], [4]. The first MAP decoder takes as input the noisy systematic information sequence $r(s)$ and the noisy parity sequence $r^{(1)}$ generated by the first recursive systematic encoder of turbo encoder. An interleaved version of and the noisy parity sequence i.e $\pi(r^{(s)})$ and $r^{(2)}$ is fed to the second MAP decoder as shown in Fig 5.

The first MAP decoder [8] then produces a soft output called extrinsic information which can be defined as

$$L_e^{\ t} = \left[\delta_0\ \delta_1.....\delta_l...\delta_{k-1}....\right]$$

Where $\delta_l = \log \dfrac{p(u_l = 1)}{p(u_l = 0)}$ and t: iteration number

Given the extrinsic information the probabilities of 1's and 0's can be computed as

$$p(u_l = 0) = \frac{1}{1+e^{\delta_l}} \text{ and } p(u_l = 1) = \frac{e^{\delta_l}}{1+e^{\delta_l}} \quad (1)$$

This extrinsic output of the first MAP decoder is interleaved in the same fashion as was by the interleaver used in the turbo encoder and is fed to the second MAP decoder where it acts as a priori information for the input information fed.The second MAP decoder[3] also produces extrinsic information which is deinterleaved(in the same fashion as was done by turbo encoder) and then fed to the first decoder to improve the estimate of a priori probabilities for its input information sequence in the next iteration. This completes one iteration. After a defined number of iterations, hard decision is done on the log likelihood ratio's values (i.e. LLR's) for the estimate of the input message.

3. MAP OR BCJR DECODER

3.1 Algorithm

The MAP algorithm is based on taking a decision based on a posteriori probability [2] of each data bit of noisy received sequence and the data bit value having the maximum a posteriori values is chosen. For an input $\underline{u} = \left[u_0\, u_1.....u_{k-1}\right]$ when passed through a rate ½ zero terminated convolutional encoder, output codeword is $\underline{v} = \left[v_0^{(1)} v_0^{(2)}\ v_1^{(1)} v_1^{(2)}.....v_{k-1}^{(1)} v_{k-1}^{(2)} v_k^{(1)} v_k^{(2)}\ v_{k+1}^{(1)} v_{k+1}^{(2)}\right]$ This codeword is then BPSK modulated which is represented as

$\underline{y} = \left[y_0^{(1)} y_0^{(2)}\ y_1^{(1)} y_1^{(2)}.....y_{k-1}^{(1)} y_{k-1}^{(2)} y_k^{(1)} y_k^{(2)}\ y_{k+1}^{(1)} y_{k+1}^{(2)}\right]$ The noisy version of transmitted signal which is finally received and needed to be decoded to recover original input signal is represented as

$$\underline{r} = \left[r_0^{(1)} r_0^{(2)}\ r_1^{(1)} r_1^{(2)}.....r_{k-1}^{(1)} r_{k-1}^{(2)} r_k^{(1)} r_k^{(2)}\ r_{k+1}^{(1)} r_{k+1}^{(2)}\right]$$

With the received value at each l^{th} bit time or stage i.e r_l the corresponding input value u_l is a random variable at the input of the receiver. So goal in bitwise MAP decoder is to compute $p(u_l|r_l)$.This probability [3] is not computed directly but instead in the form of Log Likelihood ratios defined as

$$LLR(u_l) = L(u_l = 1 \mid \underline{r}) = \ln \frac{p(u_l = 1 \mid \underline{r})}{p(u_l = 0 \mid \underline{r})} \quad (2)$$

Where

$$p(u_l = 1 \mid \underline{r}) = \sum_{s' \xrightarrow[u_l=1]{} s} p(s_l = s', s_{l+1} = s, \underline{r})$$

$$p(u_l = 0 \mid \underline{r}) = \sum_{s' \xrightarrow[u_l=0]{} s} p(s_l = s', s_{l+1} = s, \underline{r})$$

Here $s' \xrightarrow[u_l]{} s$ labels the transition from the l^{th}state to the next state i.e $(l+1)^{th}$.The received vector $\underline{r}$ has both systematic and parity bits. So the received bit $\underline{r_l}$ at any stage l can be written as $\underline{r_l} = \left[r_l^{(1)}\ r_l^{(2)}\right]$.The received sequence $\underline{r}$ can then be written in the form of past, present and future sequences values with reference to $\underline{r_l}$ as $\underline{r} = [\underline{r}_0^{l-1}, \underline{r}_l, \underline{r}_{l+1}^{k+1}]$. Then

$$p(s', s, \underline{r}) = p(s', s, \underline{r}_0^{l-1}, \underline{r}_l, \underline{r}_{l+1}^{k+1}) \quad (3)$$

Using Bayes rule the above equation is decomposed as

$$p(s', s, \underline{r}) = p(\underline{r}_{l+1}^{k+1} \mid s', s, \underline{r}_0^{l-1}, \underline{r}_l)\, p(s', s, \underline{r}_0^{l-1}, \underline{r}_l) \quad (4)$$

Ignoring past terms the above probability can be decomposed further and further as

$$p(s', s, \underline{r}) = p(s_l = s', \underline{r}_0^{l-1})\, p(s_{l+1} = s, \underline{r}_l \mid s_l = s')\, p(\underline{r}_{l+1}^{k+1} \mid s_{l+1} = s) \quad (5)$$

Now defining each of the three factors as

Forward metric: $\alpha_l(s) = p(s_{l+1} = s, \underline{r}_0^{l})$

Branch metric: $\gamma_l(s', s) = p(s_{l+1} = s, \underline{r}_l \mid s_l = s')$

Backward metric: $\beta_l(s) = p(\underline{r}_{l+1}^{k+1} \mid s_{l+1} = s)$

Then (5) can be represented in the form of product of above three probability terms as

$$p(s', s, \underline{r}) = \alpha_{l-1}(s')\gamma_l(s', s)\beta_l(s) \quad (6)$$

So LLR calculation of (2) can then be written as

$$LLR(u_l) = \log \frac{\sum_{(s',s):input=1} \alpha_{l-1}(s')\gamma_l(s',s)\beta_l(s)}{\sum_{(s',s):input=0} \alpha_{l-1}(s')\gamma_l(s',s)\beta_l(s)} \quad (7)$$

3.2. Branch Metrics

The branch metrics defined in (6) can be written as

$$\gamma_l(s',s) = p(s_{l+1} = s \mid s_l = s')\, p(r_l \mid s_l = s', s_{l+1} = s) \quad (8)$$

The first probability term is the probability of input which causes a transition from state s' to s. Eventually this probability term can be regarded as the a priori probability. Since both the value of 1 and 0 are equal likely, this is equal to ½.

$$p(s_{l+1} = s \mid s_l = s') = p(u_l = u(s' \to s)) = 1/2 \quad (9)$$

Also $r_l = \left[r_l^{(1)}\ r_l^{(2)} \right]$

Then

$$p(r_l \mid s_l = s', s_{l+1} = s) = p(r_l^{(1)}, r_l^{(2)} \mid y^{(1)}(s' \to s), y^{(2)}(s' \to s)) \quad (10)$$

Now $r_l^{(1)}$ and $r_l^{(2)}$ is Gaussian distributed with mean $y^{(1)}(s' \to s)$ and $y^{(2)}(s' \to s)$ so the branch metrics can finally be represented as

$$\gamma_l(s',s) = p(u_l = u(s' \to s)) \frac{1}{2\pi\sigma^2} e^{\frac{-\left[\left(r_l^{(1)} - y^{(1)}\right)^2 + \left(r_l^{(2)} - y^{(2)}\right)^2\right]}{2\sigma^2}} \quad (11)$$

3.3 Forward Metrics

The forward metrics[4] defined in (6) can be written as

$$\alpha_l(s) = \sum_{s'} p(s_{l+1} = s, s_l = s', \underline{r}_0^{l-1}, r_l) \quad (12)$$

Using the Bayes rule this can be written as

$$\alpha_l(s) = \sum_{s'} \gamma_l(s',s)\alpha_{l-1}(s') \quad (13)$$

This can be pictorially represented as in Fig 6.

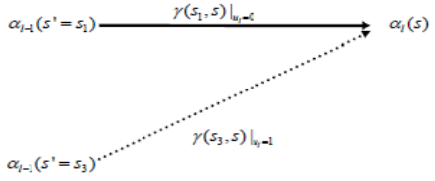

Fig. 6: Forward Metric Computation

3.4. Backward Metrics

The backward metrics defined in (6) will follow

$$\beta_{l-1}(s') = p(r_l^{k+1} \mid s_l = s') \quad (14)$$

Using Bayes rule and ignoring terms containing past information this can be represented as

$$\beta_{l-1}(s') = \sum_s \gamma(s',s)\beta_l(s) \quad (15)$$

This can be pictorially represented as in Fig.7

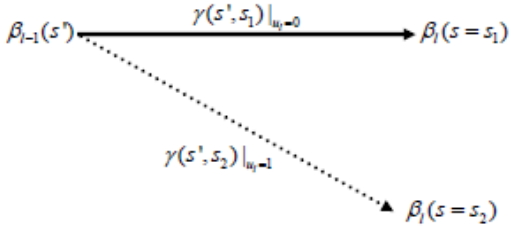

Fig. 7: Backward Metrics Computation

4. ITERATIVE MAP DECODING OF TURBO CODE

1. In the first iteration the decoder assumes all the bits are equally probable i.e. the probability of getting either 1 or 0 is ½. So $\delta_l = 0$ (for each bit) i.e. $L_{e21}{}^{(1)}(u_l) = 0$
2. For iterations t=1,2,….I where I is the total number of iterations

For 1st MAP decoder:

- Compute Output Likelihood Ratios of decoder 1 using (7). This LLR can then be represented in form of sum of three terms as

$$LLR1^{(t)}(u_l) = \pi^{-1}(L_{e21}{}^{(t-1)}) + \frac{2}{\sigma^2} r_l^{(s)} + L_{e12}{}^{(t)}(u_l) \quad (16)$$

- Compute extrinsic information output from decoder 1 for this iteration i.e $L_{e12}{}^{(t)}$

$$L_{e12}{}^{(t)}(u_l) = LLR1^{(t)}(u_l) - \pi^{-1}(L_{e21}{}^{(t-1)}) + \frac{2}{\sigma^2} r_l^{(s)} \quad (17)$$

Where $\frac{2}{\sigma^2} r_l^{(s)}$: intrinsic component

- The extrinsic information output from decoder 1 i.e $L_{e12}{}^{(t)}$ is passed through an interleaver and fed to decoder 2. So second decoder gets $\pi(L_{e12}{}^{(t)})$ along with inputs $\pi(r^{(s)})$ and $r^{(2)}$

For 2nd MAP decoder

- Compute Output Likelihood Ratios of decoder 2 using (7). Again this LLR values can be represented as sum of three terms

$$LLR2^{(t)}(u_l) = \pi(L_{e12}{}^{(t)}) + \frac{2}{\sigma^2}\pi(r_l^{(s)}) + L_{e21}{}^{(t)}(u_l) \quad (18)$$

- Compute extrinsic information output from decoder 2 for this iteration i.e $L_{e21}{}^{(t)}$

$$L_{e21}{}^{(t)}(u_l) = LLR2^{(t)}(u_l) - \pi(L_{e12}{}^{(t)}) + \frac{2}{\sigma^2}\pi(r_l^{(s)}) \quad (19)$$

Where: $\frac{2}{\sigma^2} r_l^{(s)}$: intrinsic component

1) After I number of iterations make a hard decision on the deinterleaved values of the log likelihood ratios of the second decoder to estimate the input

$$u_l = 1 \text{ for } \pi^{-1}(LLR2(u_l)) > 0 \quad (20)$$

$$u_l = 0 \text{ for } \pi^{-1}(LLR2(u_l)) < 0 \quad (21)$$

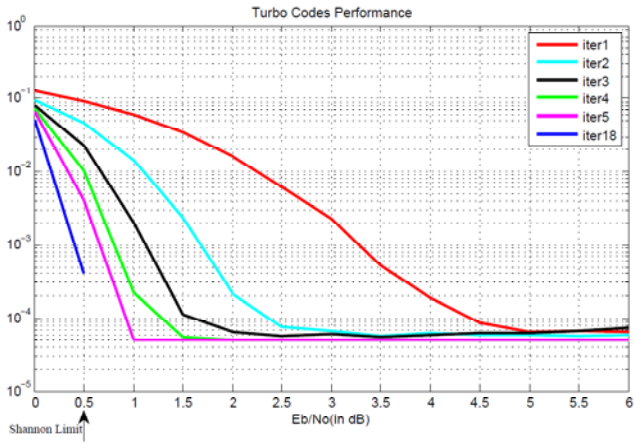

Fig. 8: Simulated BER performance of Turbo Codes with increasing iterations

With the above algorithm the BER performance of half rate Turbo decoder with increasing number of iterations can be simulated as in Fig. 8. The simulation was done in MATLAB for an encoder of constraint length 3 and and pseudo random interleaving.

As can be seen the error reduces with increasing number of iterations and consequently the performance approximates to that of the Shannon capacity [4]. So each iterative cycle produces a change adding up to a big improvement in the performance and this is what the turbo principle idea is.

5. PERFORMANCE ANALYSIS UNDER DIFFERENT SITUATIONS

5.1 Encoding rate of Turbo Code

One way of increasing the rate of turbo code is by puncturing the encoder outputs of the basic 1/3 rate turbo code. To find the optimal puncturing pattern the decoder is tested with different puncturing patterns and the one giving best BER performance is then selected.

The puncturing method most commonly done to obtain ½ code rate from 1/3 code rate decoder is to drop odd numbered parity bits from the first recursive systematic convolutional encoder and to drop even numbered parity bits of the second recursive systematic convolutional encoder as shown in the Fig 9.

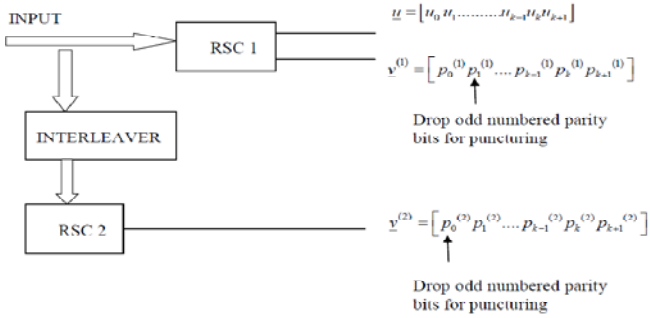

Fig. 9: Puncturing of Turbo Codes

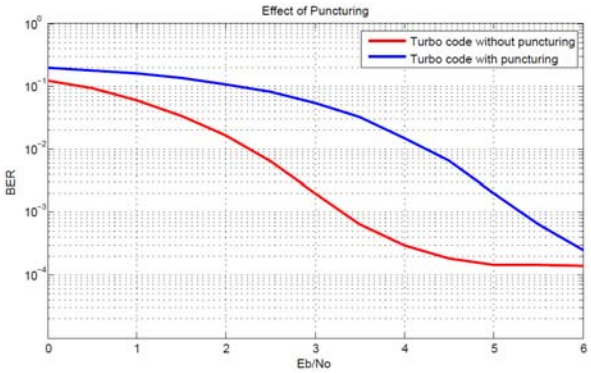

Fig. 10: Influence of puncturing on Turbo codes performance

The decoder remains the same and the punctured or missing bits are plugged in as zero by the decoder. The plot of Fig.10 shows the effect of puncturing. The simulation was done in MATLAB for second iteration for a frame size of 6000 bits with Random interleaving.

5.2 Interleaver Types

Generally the interleavers that are used of the pseudo random type or block[8] interleaving type. A block interleaver[3] formats the input sequence of N bits in a matrix of m rows and

n columns such that N=m*n. The input sequence is written into the matrix row wise and read out column wise. The deinterleaver stores the data into a matrix with the same format as interleaver. The data are read out and delivered to the decoder row wise. The performance of turbo decoder with pseudorandom interleaving as compared to block interleaving is shown in the Fig 11.The simulation was done in MATLAB for third iteration of a 1/3 rate Turbo decoder with an input frame size of 6000 bits.

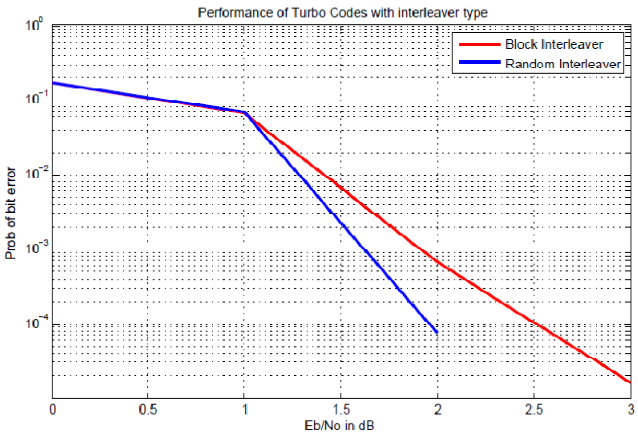

Fig. 11: Performance of Turbo Codes with interleaver pattern

5.3. Constraint Length of Convolutional Encoder

Under the same length of information sequence and the code rate, the performance of the turbo code varies with the constraint lengths of the encoder structure. This is displayed in Fig. 12.

The simulation was done for third iteration of a half rate Turbo decoder with an input frame size of 6000 bits. Random interleaving was used. The performance improvement displayed in fig is quite evident as with the increase in constraint length of the encoder, the number of soft decision levels also increase and thus the soft decision decoding is also better.

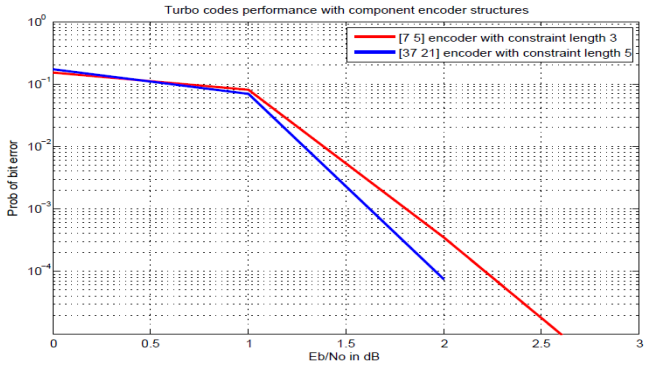

Fig. 11: Influence of constraint length on Turbo codes performance

6. CONCLUSION

Turbo code performance has been analyzed under different circumstances like number of iterations, encoding rate, and constraint length and interleaver structures. This can provide a reference to the choice of parameters for Turbo codes in hardware implementation.

REFERENCES

[1] C.Bern, A.Galvieux, and P.Thitimajshima, “Near Shannon limit error correcting coding and decoding:Turbo-code”,Proceedings of the ICC’93, pp.1064-1070, Geneva,Switzerland,May 1993

[2] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, “Optimal decoding of linear codes for minimizing symbol error rate”, IEEE Transaction on Information Theory, vol.-20,pp. 284-287, Mar. 1974

[3] Branka Vucetic, Jinhong Yuan, “Turbo Codes Principles and Applications”, Kluwer Academic Publishers, 2000

[4] Bernard Sklar, “Digital Communications-Fundamentals and Applications”, Pearson Education, Second edition, 2003

[5] Claude Berrou, Alain Glavieux, “Near Optimum Error Correcting coding and decoding: Turbo Codes”, IEEE Transactions on Communications, Vol. 44, No.10, October 1996

[6] Dhouha Kbaier Ben Ismail, Catherine Douillard, Sylvie Kerouedan, “A survey of three dimensional Turbo Codes and recent performance enhancements”, EURASIP Journal on Wireless Communications and Networking 2013

[7] J.B. Anderson, S.M. Hladik, “MAP Tailbiting Decoders”, 1997 IEEE

[8] Ramesh Mahendra Pyndiah, “Near-Optimum Decoding of Product Codes: Block Turbo Codes”, IEEE Transactions On Communications, Vol. 46, No. 8, August 1998

[9] C.E Shannon, “Communication in the presence of Noise,” Proc. IRE, Vol. 37, no.1,pp. 10-21,Jan 1949