Performance Analysis of Log-map, SOVA and Modified SOVA Algorithm for Turbo Decoder

Mohammad Salim R.P. Yadav S.Ravi kanth
Dept. of Electronics & Communication Engineering, MNIT, Jaipur, Rajasthan (India)

ABSTRACT
In this paper we analyze the BER performance of the log-map and SOVA decoding algorithms for turbo codes over the AWGN and the fading channels, the Rayleigh, the Rician and the Nakagami-m. Also a modification to the SOVA algorithm is proposed to improve its BER performance. Simulation results show that with the proposed modification to the SOVA the BER performance of SOVA is improved. Simulations were done using MATLAB.

I . INTRODUCTION

Turbo codes were first introduced in 1993 by Berrou, Glavieux, and Thitimajshima, and reported in [1,2], where a scheme is described that achieves a bit-error probability of 10^{-5} using a rate 1/2 code over an additive white Gaussian noise (AWGN) channel and BPSK modulation at an E_b/N_0 of 0.7 dB[9]. The turbo-codes invented by Berrou *et al*. should more formally be described as *parallel-concatenated recursive systematic convolutional codes.* The codes are constructed by using two or more component codes on different interleaved versions of the same information sequence. Whereas, for conventional codes, the final step at the decoder yields hard-decision decoded bits (or, more generally, decoded symbols),for a concatenated scheme such as a turbo code to work properly, the decoding algorithm should not limit itself to passing hard decisions among the decoders.

To best exploit the information learned from each decoder, the decoding algorithm must effect an exchange of soft decisions rather than hard decisions. For a system with two component codes, the concept behind turbo decoding is to pass soft decisions from the output of one decoder to the input of the other decoder, and to iterate this process several times so as to produce more reliable decisions.

With the introduction of binary turbo codes near optimum error correction became possible. Due to these error correction capabilities, binary and duo-binary turbo codes allow for low frame error rates (FER) at a low signal-to-noise ratio (SNR), outperforming the widely used convolutional codes. Because of this advantage turbo codes are now part of a large number of communication standards.

II. DECODING OF TURBO CODES

Figure 1. below shows the various decoding algorithms available for decoding of turbo codes. All the algorithms are based upon the trellis-based estimation. The trellis based estimation algorithms are classified into two types. They are sequence estimation algorithms and symbol-by-symbol estimation algorithms. The viterbi algorithm, SOVA (soft output viterbi algorithm) and improved SOVA are classified as sequence estimation algorithms. Where as the MAP algorithm, Max-Log-Map and the Log-Map algorithm are classified as symbol-by-symbol estimation algorithms. In general the symbol-by-symbol estimation algorithms are more complex than the sequence estimation algorithms but their BER performance is much better than the sequence estimation algorithms[8].

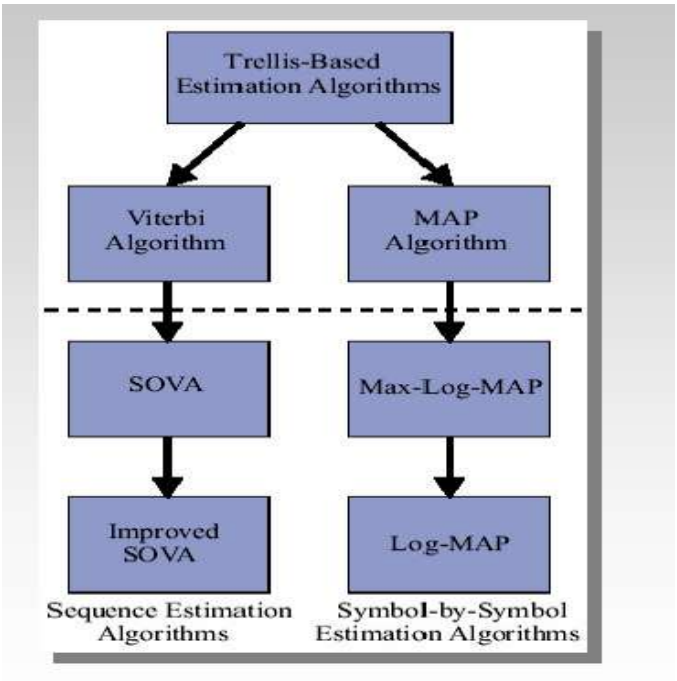

Fig.1. Decoding algorithms for turbo codes

The MAP, SOVA, LOG-MAP, MAX-LOG-MAP, improved SOVA , all these algorithms produce soft-outputs. The viterbi algorithm is a hard-decision output decoding algorithm. SOVA is soft-output producing viterbi algorithm.

III. IMPROVED SOVA

It is known that the performance of a SOVA (soft output Viterbi algorithm) turbo decoder can be improved, as the extrinsic information that is produced at its output is over optimistic. A new parameter associated with the branch metrics calculation in the standard Viterbi algorithm is introduced that affects the turbo code performance. Different parameter values show a simulation improvement in the AWGN channel as well as in an uncorrelated Rayleigh fading channel.

There are different efficient approaches proposed to improve the performance of soft-output Viterbi algorithm (SOVA)-based turbo decoders. In the first approach, an easily obtainable variable and a simple mapping function are used to compute a target scaling factor

to normalize the extrinsic information output from turbo decoders. The scaling factor can be a variable scaling factor or a fixed scaling factor.

(1) Scaling factor method:

Variable scaling factor method:

In this method [5] a scaling factor 'c' of

$$c = \mu_v\left(\frac{2}{\sigma_v^2}\right)$$

should be employed to normalize the soft output of SOVA decoders. Where μ_v and σ_v represent the mean and variance of v and v denotes the soft output to be concerned.

In practice, to compute the mean and variance of the soft output from SOVA decoders, multiplication and addition operations must be performed at each symbol-processing cycle within each iterative decoding. Also, to compute the final scaling factor, a division operation must be performed before the next iteration begins. All of these imply that a practical SOVA-based turbo decoder with the normalization process embedded may work either with a larger clock cycle period or with a considerable extra latency when pipeline techniques are employed .

Fixed scaling factor method:

In general, the input of a SISO decoder is fed by the a priori information L_{in} of information bits and the received values L_c y corresponding to coded bits, and it produces a soft output (LLR) that is the estimate of the transmitted sequence of bits. The extrinsic information L_e of the information bits can be found as

$$L_e = LLR - (L_{in} - L_e y)$$

This is then used as the new a priori information for the next decoder and the process is continued iteratively. In this method of implementation, the extrinsic information should be multiplied by a proper stability factor at every iteration step to avoid increasing BER at very low SNR.

By trial and error, we need to find the best value to multiply the SOVA extrinsic information sequence in order to correct or improve the turbo decoder performance[7].

The advantage of fixed scaling factor method over the variable scaling factor method is the reduction in computation complexity.

(2) Thresholding method:

In this approach, adaptive thresholding[5,6], a fixed upper bound is set in computing metric difference between the survivor path and the competing path in order to compensate for the overestimation of conventional SOVA.

Using this approach alone does not help much in turbo decoding performance in general cases. In fact, due to the moderate performance of conventional SOVA-based turbo decoders, the computed reliability values for most symbols are relatively small unless the SNR in the received data is really high. In other words, the upper bound set by the preset threshold is seldom reached in computing the metric difference between competing paths. Lowering the bound in these cases will increase the thresholding rate. However, the dynamic range of the extrinsic information will be limited, and the resultant algorithm will have trouble in distinguishing the relative reliabilities of various bits. So the performance improvement is not guaranteed.

IV. PROPOSED MODIFIED SOVA

The modification is with respect to the fixed scaling factor method mentioned above. The soft output of the decoder consists of three terms and is given by

$$L(\hat{d}) = L_c(x) + L(d) + L_e(\hat{d})$$

Where, $L_c(x)$ is the channel reliability factor, $L(\hat{d})$ is the priori information of the data bit, $L_e(\hat{d})$ is the extrinsic information gleaned from the decoder.The initial inherent strong correlation between the intrinsic information (input to the SOVA) and extrinsic information (output of the SOVA) leads to exaggerated extrinsic information**.** To overcome this strong correlation we scale the extrinsic information of the second decoder. As we increase the number of iterations we get a more reliable soft output of the decoder. The extrinsic information needs to be increased in the last iteration or after first few iterations to get a more reliable soft output of the decoder. So instead of using the scaling factor for all iterations, we use the scaling factor for first few iterations and then remove or increase the scaling factor in the last iteration.

V. FADING CHANNELS

We consider the three fading channels , the Rayleigh , the Rician and the Nakagami-m. The probability density functions of these channels are defined below.

RAYLEIGH FADING:

The Rayleigh fading model is one of the most widely used fading channel model which assumes that there exist no direct line of sight path between the transmitter and the receiver and all the arriving signals at the receiver are due to reflected waves. The normalized Rayleigh distribution, its mean and variance are as given below

$$P(a) = 2a\exp(-a^2), \quad a \geq 0$$
$$= 0, \quad a < 0$$

$$m_a = 0.8862, \quad \sigma_a^2 = 0.2146$$

RICIAN FADING:

In many propagation scenarios, there exist a line of sight component having constant amplitude and a number of reflected waves. The sum of direct path and the reflected components result in a signal having Rician envelope distribution. The normalized Rician distribution along with its mean and variance are

$$P(a) = \begin{cases} 2a(1+K)\exp[-K-(1+K)a^2]\, I_o\left(2a\sqrt{K(1+K)}\right), & a \geq 0 \\ 0 & a < 0 \end{cases}$$

$$m_a = \frac{1}{2}\sqrt{\frac{\pi}{1+K}}\exp\left(-\frac{-K}{2}\right)\left[(1+K)I_0\left(\frac{K}{2}\right) + KI_1\left(\frac{K}{2}\right)\right]$$

$$\sigma_a^2 = 1 - m_a^2$$

In the above expressions *K* is the Rician factor and it represents the power ratio of the direct and reflected signal components. Additionally, $I_0(.)$ and $I_1(.)$ represent the modified Bessel functions of first kind having zero and first order. Small values of *K* imply severe fading whereas, large values indicate mild fading. When *K* = 0, the Rician pdf becomes the well known Rayleigh pdf whereas, in the case of K ->∞ it corresponds to the Gaussian channel.

NAKAGAMI-m FADING:

The Nakagami-*m* fading model is another widely used channel model for fading environments and the Nakagami factor *m* is the shape parameter which controls the severity of amplitude fading. The justification for the use of the Nakagami-*m* fading model is due to its good fit to empirical fading data. The normalized Nakagami-*m* distribution with its mean and variance is as shown below

$$P(a)= \begin{cases} \frac{2m^m a^{2m-1}}{\Gamma(m)} e^{-ma^2}, & a \geq 0, m \geq \frac{1}{2} \\ 0 & a < 0 \end{cases}$$

$$m_a = \frac{\Gamma(m+\frac{1}{2})}{\Gamma(m)\sqrt{m}}$$

$$\sigma_a^2 = 1 - \frac{1}{m}\left[\frac{\Gamma(m+\frac{1}{2})}{\Gamma(m)}\right]^2$$

where Γ(.) denotes the gamma function. The value *m* = 1, results in the most widely used Rayleigh fading model. Values of *m* less than unity correspond to fading more severe than Rayleigh fading, whereas values greater than one represent milder fading effects.

The Rician distribution can be approximated by the Nakagami-*m* distribution by interchanging Nakagami fading parameter *m* by the following relation

$$m= \left[1 - \frac{K^2}{(1+K)^2}\right]^{-1} \text{-(1)}$$

VI. SIMULATION RESULTS

The simulation results are as shown in figures 1,2,3 and 4. Figure 1 shows the BER performances of the SOVA and log-map decoding algorithms over AWGN (additive white Gaussian noise channel). We can observe that the log-map algorithm outperforms the SOVA algorithm.

Figure 2 shows the BER performance of SOVA over Rayleigh, Rician and nakagami-m fading channels. We considered the rician factor k=1. For k=1 we get the value of m greater than 1 from the above relation (1). So we can observe that the BER performance is worst over Rayleigh channel than Rician than nakagami-m. Also the performance degraded over fading channels when compared to the AWGN channel.

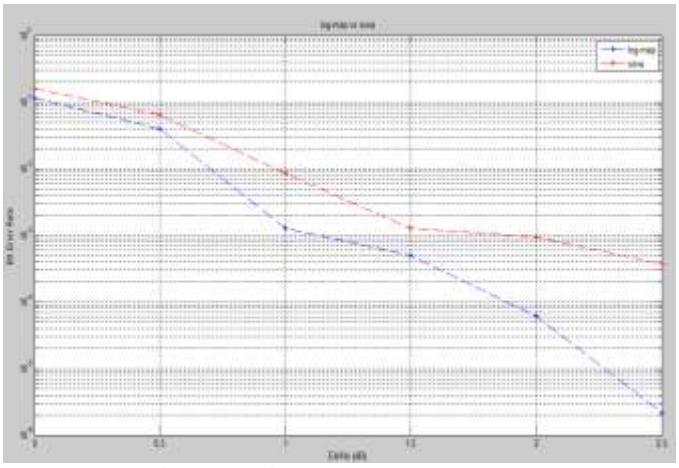

Fig.1. BER performance of log-map and sova over AWGN channel.

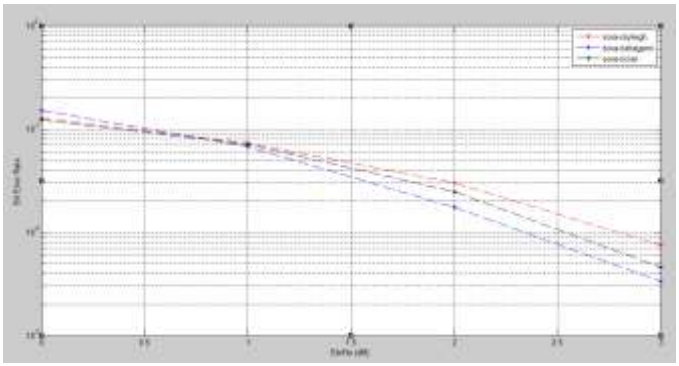

Fig.2. BER performance of SOVA over Rayleigh, rician and nakagami-m fading channels.

Figure 3 below shows the BER performance of log-map over Rayleigh, rician and nakagami-m fading channels. Here also we can observe that the performance of log-map degraded over fading channel when compared to the AWGN channel. Also the performance of log-map is worst over Rayleigh than rician than nakagami-m. also from fig 2 and 3 we can observe that even for fading channels the performance of log-map is superior when compared to the SOVA .

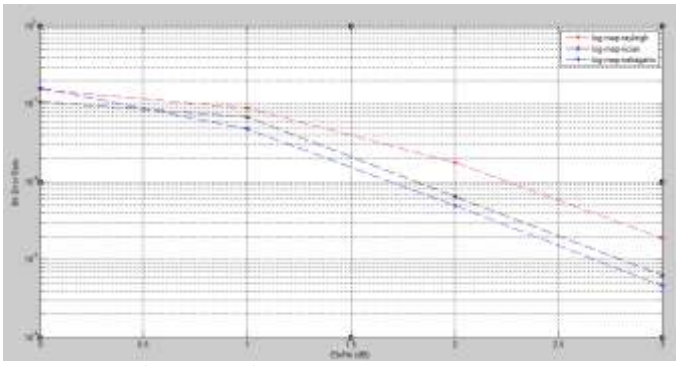

Fig.3.BER performance of log-map over Rayleigh, rician and nakagami-m fading channels.

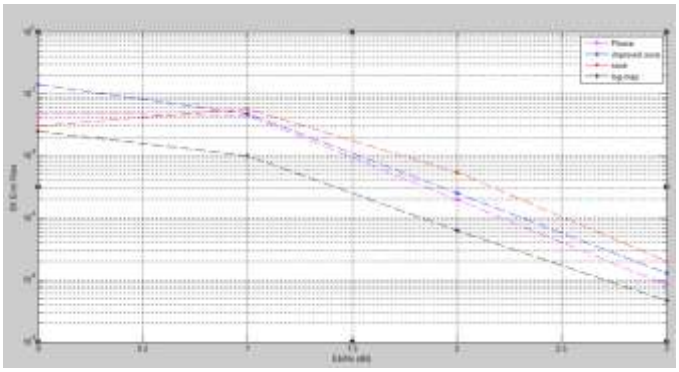

Fig.4. BER performance of log-map, sova, improved sova and further improved sova.

From figure 4 above which shows the BER performance of log-map, SOVA, improved SOVA, and further improved SOVA (FIsova). Improved SOVA is with the help of scaling factor of 0.73. Further improved SOVA (FIsova) is the modification suggested above . We applied scaling factor of 0.72 for first four iterations and then removed the scaling factor for the last iteration. The total number of iterations is 5. So we can observe that using a fixed scaling factor of 0.72 there is an improvement in SOVA which is the improved SOVA and further removing the scaling factor for the last iteration we can see that there is further improvement (FIsova) as named in the figure 4 above.

VII. CONCLUSION

We analyzed the performance of log-map and SOVA over the AWGN and fading channels. We observed that for both fading channels and the AWGN channels the performance of log-map algorithm is superior as compared to the SOVA. It has been verified by the simulation results that the proposed modification to the fixed scaling factor method gives improved results of the performance of SOVA. We observe that removing the scaling factor for the last iteration improves the performance of SOVA algorithm further. These results will be helpful in designing Turbo codes with reduced decoding latency particularly in fourth and higher generation mobile communication systems.

REFERENCES

1. C. Berrou, A. Glavieux, and P. Thitimasjshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes," Proceedings of the IEEE International Conference on Communications, pp. 1064–1070, May 1993.
2. C.Berrou, and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo-Codes," *IEEE Transactions on Communications*, vol. 44, no. 10, pp. 1261-1271, October 1996.
3. Joachim Hagenauer and Lutz Papke , "Iterative Decoding of Binary Block and Convolutional Codes", *IEEE Transactions on Information Theory*, vol. 42, No. 2, pp. 429-445, March 1996.
4. **J.** Hagenauer and L. Papke, "Decoding 'Turbo' codes with the soft output Viterbi algorithm (SOVA)," in Proc. *Int.* Symp. ***on*** Information Theory (Trondheim,Norway, June 1994), p. 164.
5. Zhongfeng Wang, and Keshab K. Parhi, "High Performance, High Throughput Turbo/SOVA Decoder Design", *IEEE Transaction on Communications*, Vol. 51, No. 4, pp. 570-579, April 2003.
6. S. Papaharalabos, P. Sweeney and B.G. Eva,"Modification of branch metric calculation to improve iterative SOVA decoding of turbo codes"electronics letters 18th September 2003 Vol. 39 No. 19.
7. T.Gnanasekaran and Dr.K.Duraiswamy,"Application of Scaling factors for MAP and SOVA for Robust Performance in Forward Error Correction" International Journal of Recent Trends in Engineering, Vol 1, No. 3, May 2009.
8. Jinhong Wu and Branimir R. Vojcic,"Combining Iterative SOVA and Log-MAP Algorithms for Turbo Decoding"2009 IEEE,43rd annual conference, information sciences and systems.
9. Bernard Sklar, "*Digital Communications: Fundamentals and Applications*", Pearson Education, Second Edition, 2006.