DOI: 10.1515/SBEEF-2019-0009

PERFORMANCE ANALYSIS OF TURBO CODES OVER AWGN CHANNEL

MOHANAD ABDULHAMID[1], MBUGUA THAIRU[2]
[1]AL-Hikma University, Iraq, [2]University of Nairobi, Kenya
E-mail: moh1hamid@yahoo.com, researcher12018@yahoo.com

Abstract. ***Turbo coding is a very powerful error correction technique that has made a tremendous impact on channel coding in the past two decades. It outperforms most known coding schemes by achieving near Shannon limit error correction using simple component codes and large interleavers. This paper investigates the turbo coder in detail. It presents a design and a working model of the error correction technique using Simulink, a companion software to MATLAB. Finally, graphical and tabular results are presented to show that the designed turbo coder works as expected.***

Keywords: performance analysis, turbo codes, AWGN channel

1. INTRODUCTION

Communication in all its forms is an incredibly crucial part of our world. Without effective and reliable communication, information systems cease to function and eventually break down. It is not surprise then that the telephone is hailed as one of the most significant inventions of the 20th century. The extensive research and theories that we spurred on by its invention have a significant way shaped the telecommunications landscape we know today.

The ability to transmit information reliably over a noisy channel at relatively low power made many innovations that were simply not practical beforehand possible. Communication devices could now be constructed smaller, satellite communication systems could be more energy efficient and information could be transmitted further, in both wired and wireless channels with lower error rates. This was achieved through the introduction of error control codes.

As one of the error control codes, turbo codes made the first big leap towards reaching the Shannon limit, the highest transmission rate that can be achieved over a noisy channel without errors. This enhanced research and innovation in the field worldwide, with over 400 patents involving its theory and applications being filed since its development. Today turbo codes are at the center of high speed wireless communication and are only rivaled in performance by the low density parity check (LDPC). Several researches which deal with turbo codes can be found in literatures [1-6].

2. DESIGN METHODOLOGY

The simulation of a turbo code is carried out in MATLAB-2016 in Simulink Version 8.7 (R2016a). The turbo coding model is made up of MATLAB blocks, user-defined MATLAB system blocks as well as user-defined functions, all of which are necessary for its functioning. The primary toolboxes used are the communication system and discrete system processing (DSP) toolbox. The model has the following specifications:

- Code rate: 1/3;
- Encoder polynomials: 13, 15;
- Interleavers: QPP and RANDOM interleavers
- Modulation: QPSK (BPSK and 8-PSK used as well);
- Frequency of source: 100kHz.

Figure 1 shows the complete turbo coding system model. The system has been broken down into four main sections, control block, transmitter, channel, and receiver.

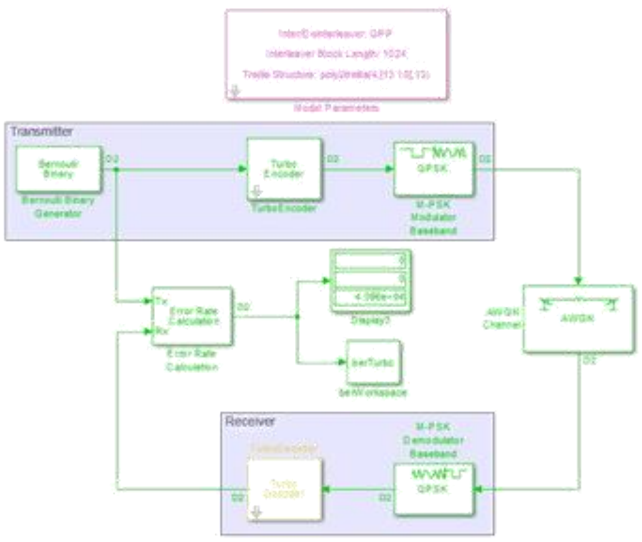
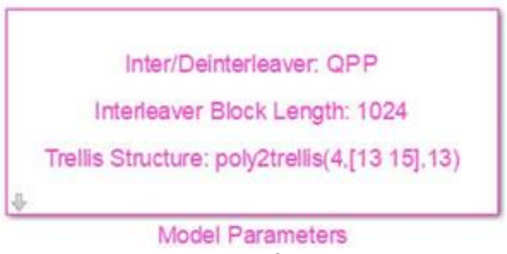


Figure 1. Block diagram of turbo coded transmission

2.1 Control block

The block shown in Figure 2 which is model parameters, is an empty port-less block that is used to set the interleaver type, interleaver block length and trellis structure for the entire model by changing its parameters. It also displays these parameters before and after a change is made. It therefore provides a central point from which to change the system parameters.



Inter/Deinterleaver: QPP

Interleaver Block Length: 1024

Trellis Structure: poly2trellis(4,[13 15],13)

Model Parameters

Figure 2. Model of control block

2.2 Transmitter

As illustrated in Figure 1, the transmitter is made up of the Bernoulli binary generator, the turbo encoder and the modulator.

2.2.1 Bernoulli binary generator

This block acts as the source of the transmission system and transmits ‘0’s and ‘1’s with equal probabilities. It is compared against the output of the turbo decoder to determine the error performance of the system. Its sample time is set to 10ms which set its frequency at 100 kHz. However, its samples per frame are dependent on the block length of the interleaver and is changed automatically when the ‘blocklength’ property of the control block is adjusted.

2.2.2 Turbo encoder

Figure 3 below shows the turbo encoder’s subsystem and its sub-blocks are explained in the subsequent sections.

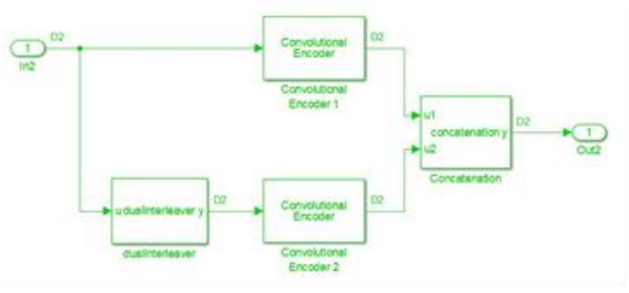


Figure 3. Turbo encoder’s subsystem

2.2.2.1 Convolutional Encoder

This is a communications system toolbox block that performs the convolutional encoding using the polytrellis function (polynomial 13,15). The function constructs the trellis required by the encoder block and thus determines the encoder type and configuration.

2.2.2.2 Dualinterleaver

This user-defined block performs either RANDOM or quadratic permutation polynomial (QPP) interleaving depending on the selection at the control block. The RANDOM interleaving is performed using the communications system toolbox’s RANDOM interleaver.

The QPP interleaver uses a table of 19 polynomial and their inverses to calculate the permutations. The table limits the block lengths that can be used to 19 values between 40 and 8192.

2.2.2.3. Concatenation

This is a user-defined block that concatenates the systematic bits with the 2 sets of parity bits from the encoders. Its parameters are obtained from the function ‘enTrellisParameters’ which retrieves the parameters of the trellis from the polytrellis object in the encoder and makes them available to all the blocks within the turbo encoder block.

The block takes in the two convolutional codewords, removes the interleaved systematic bits and concatenates its parity bits with the codeword of the upper encoder. The two sets of termination bits are then added to the end of the new (3, 1) codeword.

2.2.3 Modulator

The primary modulator used in this system is the quadrature phase shift keying (QPSK) because it has good error performance. However, binary phase shift keying (BPSK) and 8-PSK are also used to test the turbo coder’s performance when using different modulation schemes. The shifts between QPSK, BPSK and 8-PSK are performed programmatically using the function ‘berSimulation’.

2.3 Channel

An additive white Gaussian noise (AWGN) channel is used as a communication channel. Its signal-to-noise ratio (Eb/N0) values are set by the function ‘berSimulation’, while the symbol period is updated automatically by the model itself each time the simulation is run.

2.4 Receiver

As illustrated in Figure 1, the receiver is made up of the demodulator and the turbo decoder.

2.4.1 Demodulator

The demodulators used are the QPSK, BPSK and 8-PSK with preference to the former. They are set with the decision type; log-likelihood, because the decoder requires soft inputs. They however use -1 to represent a one and +1 to represent a zero, the inverse of what the decoder expects.

2.4.2 Turbo decoder

The Figure 4 illustrates the different parts of the turbo decoder. The different colors represent components with different sample rates where red is the highest rate, green is the lowest and gold represents a block that has components running with different sample rates within it.

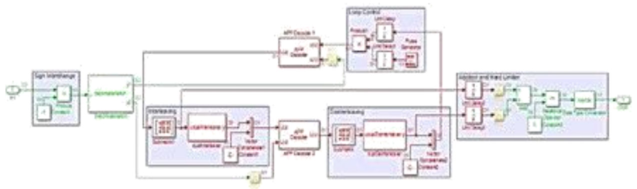


Figure 4. Turbo decoder's subsystem

2.4.2.1 Sign interchange

This block inverts the signs of the log likelihood ratio (LLRs) being received from the demodulator by multiplying the bits by '-1'.

2.4.2.2 Deconcatenation

This user-defined block performs the de-concatenation of the received noisy codeword. It separates the codeword into systematic and parity bits for A Posteriori Probability (APP) decoder 1 and parity bits alone for APP decoder 2 with their respective tail bits reattached. Its parameters are obtained from the function 'decTrellisParameters' which retrieves them from the APP decoder's polytrellis object and makes them available to the blocks within the turbo encoder block.

2.4.2.3 APP decoder

This communications system toolbox block accepts the channel information (systematic and parity bits) and a priori information and outputs an updated version of the a priori information which, in this model, is the extrinsic information. It uses the algorithms True APP, Max* and Max which are analogous to the MAP BCJR, LOG MAP BCJR and MAX LOG MAP BCJR algorithms respectively. The block parameters are entered using the turbo decoder block or by the function 'berSimulation'.

2.4.2.4 Interleaving and deinterleaving

Figure 5 and Figure 6 below show the interleaving and de-interleaving areas in the decoder. They are largely identical apart from the dualinterleaver/dualdeinterleaver block at their center. The submatrix block removes the tail bits before the interleaving is performed while the 'Vector Concatenate' block reattaches a set of zeros equal in number to the tail bits. This is to satisfy the a priori information input length constraints of the APP decoder.

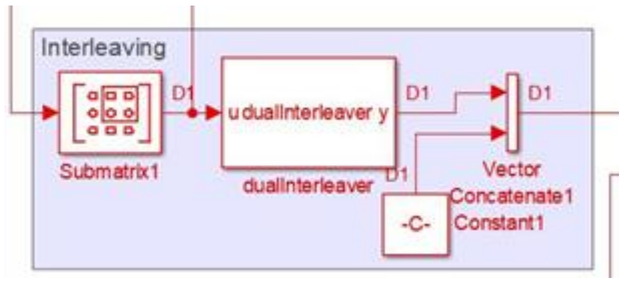


Figure 5. Block diagram of interleaving in decoder

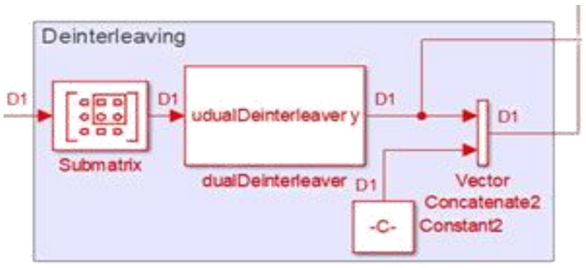


Figure 6. Block diagram of de-interleaving in decoder

2.4.2.5 Loop control

These set of blocks control the a priori information being received by the APP decoders. The 'Unit Delay' delays the set of *k* bits by one sample. This holds back the output of 'APP decoder 2' so that its first output LLR is due the output of APP decoder 1 that ran at the first sample. 'Unit Delay1' however, delays the 'Pulse Generator' to keep the output from 'APP decoder 2' synchronized with the pulses generated. Figure 7 shows the loop control of turbo decoder.

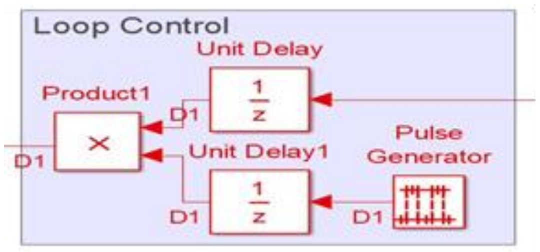


Figure 7. Loop control of turbo decoder

Figure 8 shows the signal stream from the 'Product1', 'Unit Delay' and 'Unit Delay 1' blocks for a single bit from the set of *k* bits entering each block. The letters **B, C, D** and **E** indicate the progressive step-wise increase of the a priori LLRs to the APP decoder 1 when set to 3 iterations.

The 'Pulse Generator' block controls the duration of the feedback by resetting the LLR to zero when a new set of samples is received as indicated by the letter **A** below. Each sample, **B – E** initiate new iteration. However, the final iteration initiated by **E** is ignored by the decoder, resulting in 3 instead of 4 iterations.

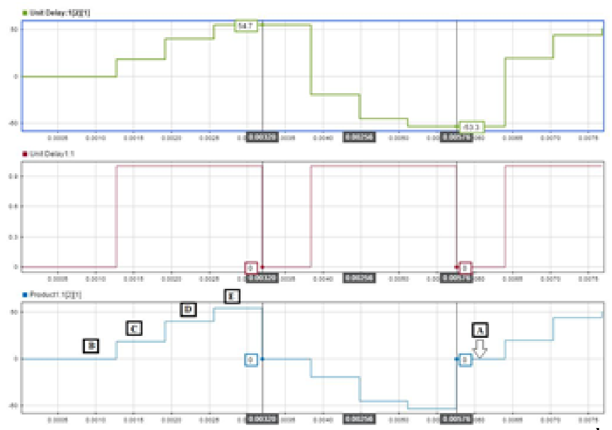


Figure 8. Control of a priori information to decoder at 2nd set of samples (between 0.00320 and 0.00576)

2.4.2.6 Addition and hard limiter

Figure 9 shows the component blocks that perform addition and make a hard decision on the LLR when the iterations end. The two delays, 'Unit Delay' and 'Unit Delay1' select the previous iteration's LLR rather than the most recent. With reference to Figure8, they allow **E** to be ignored and the 3rd's iteration LLR from 'APP Decoder 2' (**D**) to be taken.

The 'Rate Transition blocks', RT4 and RT5 interface the two sections of different sample rates. They select the final LLR at the end of the iterations (**D**) to be outputted to the 'Add' block.

The 'Add' block adds the extrinsic information going to the APP decoder 2 with the a priori information going to the APP decoder 1. This is equivalent to the de-interleaved BCJR algorithm output from decoder 2.

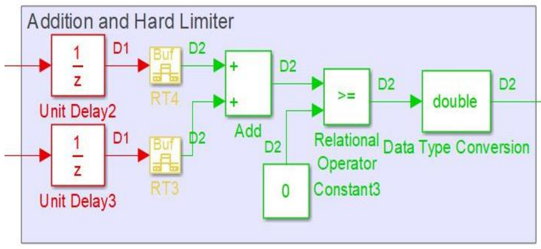


Figure 9. Addition and hard limiter blocks of the decoder

The 'Relational Operator' compares the value of the LLR with '0'. If it is greater than 0 (or positive), it outputs a 1 and if it's less than 0 (negative), it outputs a 0. In this way, a hard decision is made on the LLR.

2.5 Performance testing

There are several blocks, systems and functions used for performance testing of the turbo coding system. Figure 1 shows the 'Error Rate Correction' block which is a communications system toolbox block that compares the transmitted and received signal and calculates the bit error rate (BER) from the comparisons.

The function 'berSimulation' generates all the BER curves required. It uses the results from the error correction block, parameters entered into the function as well as a second Simulink model (Figure 10) to plot the BER curves.

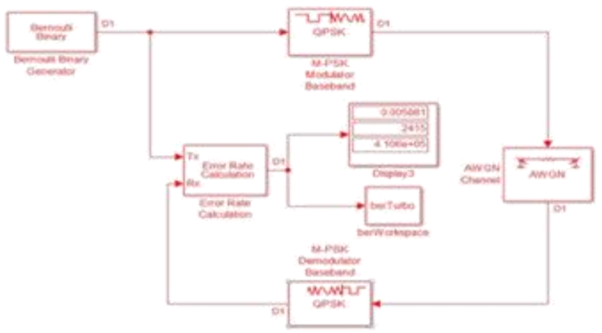


Figure 10. Uncoded transmission model

The model above is used to plot the BER curve when turbo coding isn't applied using the 'berSimulation' function when this option is chosen.

3. RESULTS

In this section, the performance of the turbo coder is illustrated and evaluated. For the turbo coder, the parameters of interest are the interleaver block length, the number of iterations, the algorithm or interleaver used. The BER performance of the system is investigated with regard to these parameters. The time elapsed during the decoding and the effect of different modulation schemes on turbo code performance are also investigated. The BER curves are plotted by varying these parameters. Each BER point on the curve is obtained by running the model through 8.192×10^6 bits. The table under each figure give the simulation time of each curve, where simulation time here can be viewed as a measure of the decoding speed of the decoder under a predetermined set of parameters.

3.1 Decoding iterations

Figure 11 below shows the decoder's error performance as the number of iterations are increased. It shows good convergence at lower Eb/N0, however it also shows a mediocre asymptotic gain. For the iteration 18 BER curve, it's about 1.25dB. Table 1 shows time elapsed when iterations are varied.

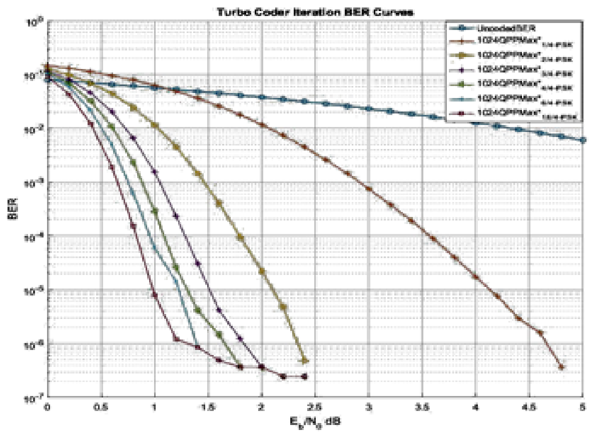


Figure 11. Performance at different iterations

Table 1. Time elapsed when iteration are varied

Turbo Coder Iteration Curves (02:00:18.035)	
1024QPPMax*1/4-PSK	00:13:18.227
1024QPPMax*2/4-PSK	00:09:59.357
1024QPPMax*3/4-PSK	00:12:32.457
1024QPPMax*4/4-PSK	00:15:02.973
1024QPPMax*6/4-PSK	00:19:57.040
1024QPPMax*18/4-PSK	00:49:05.290

3.2 Interleaver block length

The error performance when the interleaver and de-interleaver block length is varied is shown in Figure12. The error floor is shown to lower as the block length increases.

The flat result shown by the 8192 and 2048 block length curves is due to the limit to the number of bits that could be used for the simulation. A greater number, if used, would have displayed the error floor more clearly. However, it still indicates the presence of an error floor. Table 2 shows time elapsed when block length is varied.

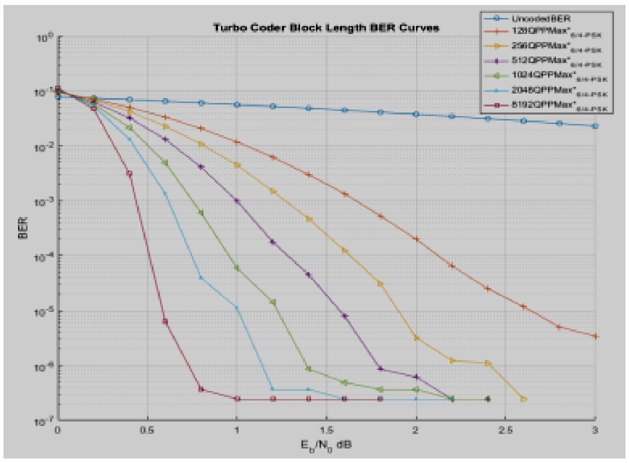


Figure 12. Performance with different interleaver block lengths

Table 2. Time elapsed when bloc length is varied

Turbo Coder Block Length Curves (02:01:52.176)	
128QPPMax*6/4-PSK	00:25:45.068
256QPPMax*6/4-PSK	00:22:40.831
512QPPMax*6/4-PSK	00:20:44.180
1024QPPMax*6/4-PSK	00:19:28.221
2048QPPMax*6/4-PSK	00:17:53.846
8192QPPMax*6/4-PSK	00:15:05.421

3.3 Decoding algorithm

Figure 13 shows the performance of the three algorithms that are available. The error and simulation time performance of the Max* algorithm indicate why it is used for generating the rest of the results. Table 3 shows time elapsed when algorithm is varied.

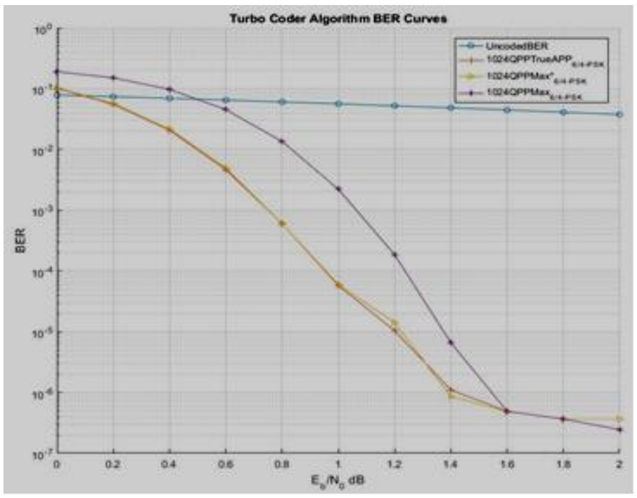


Figure 13. Performance with different algorithms

Table 3. Time elapsed when algorithm is varied

Turbo Coder Algorithm BER Curves (00:52:14.787)	
1024QPPTrueAPP 6/4-PSK	00:25:47.991
1024QPPMax*6/4-PSK	00:15:02.286
1024QPPMax 6/4-PSK	00:11:13.943

3.4 Interleaver and deinterleaver

The performance of the two integrated interleavers at low and high iteration values is shown in Figure 14. The QPP interleaver shows superior performance to the RANDOM interleaver in both the decoding speed and level of error floor. Table 4 shows time elapsed when interleaver is varied.

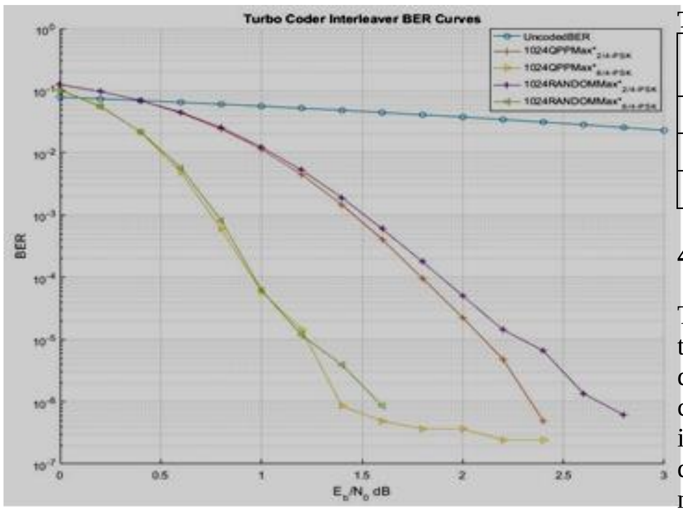


Figure 14. Performance with different interleavers

Table 4. Time elapsed when interleaver is varied

Turbo Coder Interleaver BER Curves (01:21:34.429)	
1024QPPMax*2/4-PSK	00:09:41.547
1024RANDOMMax*2/4-PSK	00:22:47.760
1024QPPMax*6/4-PSK	00:19:19.912
1024RANDOMMax*6/4-PSK	00:29:30.851

3.5 Modulation schemes

The performance of the decoder when BPSK, QPSK and 8-PSK are used can be seen in Figure 15. The BPSK and QPSK are shown to have similar performance, as expected from theory. However, the QPSK's marginally better performance at low values of Eb/N0 show why it is used as the default modulation scheme. Table 5 shows time elapsed when modulation scheme is varied.

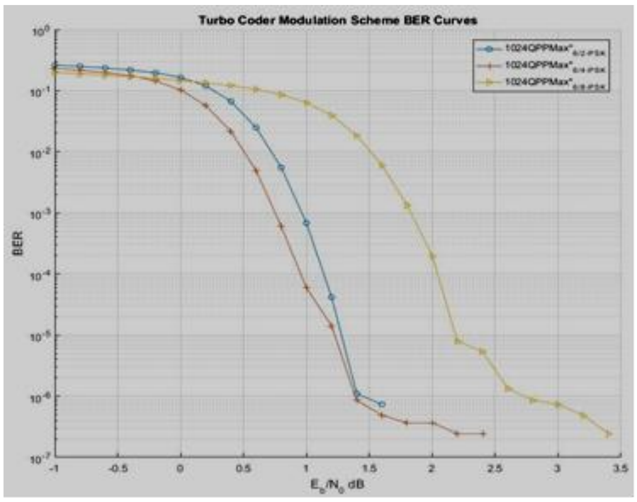


Figure 15. Effect of different modulation schemes

Table 5. Time elapsed when modulation scheme is varied

Turbo Coder Modulation Scheme BER Curves (01:22:56.186)	
1024QPPMax*6/2-PSK	00:20:41.384
1024QPPMax*6/4-PSK	00:26:34.156
1024QPPMax*6/8-PSK	00:35:40.127

4. CONCLUSION

This paper was established to investigate and describe the turbo codes in detail, to design turbo codes and to demonstrate its performance. The performance of turbo codes is measured by considering several parameters including decoding iterations, interleaver block length, decoding algorithm, interleaver and deinterleaver, and modulation schemes.

5. REFERENCES

[1] H. Sadjadpour, N. Sloane, M. Salehi, and G. Nebe, "Interleaver design for turbo codes", IEEE Journal on Selected Area in Communications, Vol.19, No.5, 2001.

[2] J. Kaza and C. Chakrabarti, "Design and implementation of low-energy turbo decoders", IEEE Transactions on Very Large Scale Integration Systems, Vol. 12. No. 9, 2004.

[3] K. Sun, D. Yuan, X. Zhou, "Performance analysis of turbo codes with different interleavers and decoding methods", 2010 IEEE International Conference on Information Theory and Information Security, China, 2010.

[4] S. Jasim, and A. Abbas, " Performance of turbo code with different parameters", Journal of Babylon University of Engineering Sciences, Vol. 25, No.5, PP.1684-1692, 2017.

[5] M. Devi, K. Ramanjaneyulu, and B. Krishna, "Performance analysis of sub-interleaver based turbo codes", Cluster Computing, Published Online First in March, 2018.

[6] B. Ahn, S. Yoon, and Jun Heo, "Low complexity syndrome-based decoding algorithm applied to block turbo codes", Access IEEE, Vol. 6, PP.26693-26706, 2018.