الجـمـهورية الجزائرية الديمقراطية الشعبية
**République Algérienne démocratique et populaire**

وزارة التعليم الــعــالي و البحــث العــلمــي
**Ministère de l'enseignement supérieur et de la recherche scientifique**


جــامعة سعد دحلب البليدة
**Université SAAD DAHLAB de BLIDA**

كلية التكنولوجيا
**Faculté de Technologie**

قسم الإلكترونيـك
**Département d'Électronique**


**Mémoire de Projet de Fin d'Etudes**

présenté par


AHMED SHUAIB ABDURAHMAN


pour l'obtention du diplôme de Master en Électronique option desystèmes de télécommunications

Thème

# ETUDES DES PERFOMANCES DES CODES TURBO

Proposé par : Professeur Meliani Hamza

Année Universitaire 2018-2019

الجـمهورية الجزائرية الديمقراطية الشعبية
**République Algérienne démocratique et populaire**

وزارة التعليم الــعــالي و البحــث العــلمــي
**Ministère de l'enseignement supérieur et de la recherche scientifique**

جــامعة سعد دحلب البليدة
**Université SAAD DAHLAB de BLIDA**

كلية التكنولوجيا
**Faculté de Technologie**

قسم الإلكترونيـك
**Département d'Électronique**

**Thesis Project Memorandum**

presented by

AHMED SHUAIB ABDURAHMAN

For attaining a Master's Degree in Electronics option of Telecommunication systems

Theme

# STUDY OF TURBOCODESPERFOMANCES

Proposed by : Professor Meliani Hamza

Année Universitaire 2018-2019

# Acknowledgements and dedications

*Foremost, I would like to express my sincere gratitude to my promotor Prof. Meliani Hamza for the continuous support of my studies as a master's student and research, for his patience, enthusiasm motivation and his great knowledge.*

*I would like to thank the jury panel for accepting to examine my thesis.Besides that, I would like to thank the head of our departmentand the rest of my teachers for their full dedication and encouragement during my course of studies.*

*I would also like to thank my family: My parents Mr. Abdurahman Ahmed and Mrs. Farida Suleman for raising me spiritually and their full support and encouragement during my entire school life.My siblings, Ahmed, Zuhura, Yasmin and Rishad for the moral and emotional support that has played a big role in my studies and the entire research of my final project.*

*Also not forgetting my uncle Sen. Hassan Omar Hassan and Capt. Muha for their full unforgettable support.*

*Lastly to all my friends, Alfred Mwambingu, Abbas Ali, all Kenyan students in Algeria, classmates and everyone who has offered support in many ways; my heartfelt gratitude*

*SHUAIB A AHMED*

تستكشف هذه الأطروحة تصميم وتنفيذ وأداء أكوادالتوربو من خلال تغيير طول الإطار وعدد التكرارات وخوارزمية فك
التشفير في وحدة فك الترميز. يتكون مشفر التوربو من مشفري تلافيفي منهجي تكراري (RSC) متصلين على التوازي

ويفصل بينهما مشذر. يتكون ناتجها من وحدات بت منتظمة (بتات معلومات) وبتات تعادل. أثناء فك التشفير ، يتلقى مفكك التشفير كلمة المرور ويحسب نسب السجل (LLRs) باستخدام إما سجل MAP أو MAX log. يتم فك التشفير التكراري على LLRs فك الشفرة من أجل إخراج أفضل. توضح المقارنة بين نتائج الخوارزميتين أن سجل خوارزمية MAP يعطي نتائج BER أفضل ولكن على حساب الوقت بينما MAX log يوفر MAP أداء أقل قليلاً مع تقليل وقت التنفيذ.

الكلمات المفتاحية: شفرة Turbo، BER، Shannon، RSC ، تشذير ، بتات تعادل ، LLRs ، فك تشفير تكراري.
طول الإطار

Ce mémoire explore la conception, la mise en œuvre et les performances des turbo-codes en faisant varier la longueur de trame, le nombre d'itérations et l'algorithme de décodage au niveau du décodeur. Un turbo-codeur est constitué de deux codeurs RSC (Récursive Systématique Convolutional) connectés en parallèle et séparés par un entrelaceur. Sa sortie est constituée de bits systématiques (bits d'information) et des bits de parité. Lors du décodage, le décodeur reçoit le mot de code et calcule les rapports Log like (LLR) à l'aide de l'algorithme log MAP ou de l'algorithme MAX log MAP. Le décodage itératif est effectué sur les LLR décodés pour une meilleure sortie. La comparaison entre les résultats des deux algorithmes montre que l'algorithme log MAP donne de meilleurs résultats de BER, mais au prix du tempsd'exécution, alors que MAX log MAP donne des performances de BER légèrement inférieures avec un temps d'exécution réduit.

Mots clés: code turbo, BER, Shannon, RSC, entrelaceur, bits de parité, LLR, décodage itératif. Longueur du cadre

<u>Abstract</u>

This thesis explores the design, implementation and performance of turbo codes by varying the frame length, the number of iterations and the decoding algorithm at the decoder. A turbo encoder consists of two Recursive Systematic Convolutional (RSC) encoders connected in parallel and separated by an interleaver. Its output consists of systematic bits (information bits) and the parity bits. During decoding, the decoder receives the codeword and computes the Log like Ratios (LLRs) using either log MAP or MAX log MAP algorithm. Iterative decoding is done on the decoded LLRs for better output. The comparison between the results of the two algorithms shows that log MAP algorithm gives better BER results but at the cost of the execution time while MAX log MAP gives slightly lower BER performance with reduced execution time.

Key words*: Turbo code, BER, Shannon, RSC, interleaver, parity bits, LLRs, iterative decoding. Frame length*

## List of Acronyms andAbbreviations

- APP: a posteriori probability
- AWGN: Additive White Gaussian Noise
- BCJR: Bahl, Cocke, Jelinek and Raviv
- BER: Bit Error Rate
- BSC: Binary Symmetric Channel
- DVB-RCS: Digital Video Broadcasting - Return Channel via Satellite
- ECC: Error Control Coding
- HSPA: High Speed Packet Access
- IM: Intermodulation
- ISI: Inter Symbol Interference
- LLRs: Log Likelihood Ratios
- LTE: Long Term Evolution
- MAP : Maximum Posteriori
- MHD: Minimum Hamming Distance
- MLD: Maximum Likelihood Decoding
- MLE: Maximum Likelihood Estimation
- PC: Parallel concatenation
- RMS: Root Mean Squared delay
- RSC: Recursive systematic convolutional
- SC: Serial concatenation
- SNR: Signal to Noise Ratio
- SOVA: Soft Output Viterbi Algorithm

- VA: Viterbi Algorithm
- WiMAX : Worldwide Interoperability for Microwave Access

## Table of Content

## List of figures

# List of tables

# General introduction

In information theory, Berrou and Glavieux developed turbo codes, which belong to a class of concatenated error correcting codes that closely approach the Shannon channel capacity, in 1993. Turbo codes at the time of their introduction were so revolutionary that researchers could not believe the results; they were later on developed and led to iterative signal processing. Concatenating smaller codes came to be a very useful tool in high rate digital communication and is thus used in modern standards [1]

The main objective of this thesis is to study the performance and the implementation of turbo codes by varying the frame length, number of iterations and the decoding algorithm used at the decoder.

Our thesis is therefore divided into four chapters;

- The first chapter is dedicated to generalities on digital communication, whereby it describes its general structure and some of its parameters.
- The second chapter introduces concatenated codes; serial and parallel concatenated codes (Turbo codes). It describes in details the convolutional codes that the convolutional turbo codes use as its encoders. It further discusses the different types of the interleaver used in the concatenation.
- The third Chapter describes the decoding of turbo codes; the iteration process along with different algorithms used at the decoder.
- The fourth chapter presents simulation results using MATLAB in terms of BER performance based on different parameters such as the effect frame length, the number of iteration processes and the kind of decoding algorithm used in the decoder.

The thesis is finally concluded by a general conclusion.

# Chapter 1: generalities on digital communication

## 1.1 Introduction

Basic principal of communication is the exchange of information between two parties. Analog communication is the transmission of analog continuous signal while Digital communication is the transfer of digital bit stream over a channel. The main objective of digital communication is the transmission of data with low energy, that can tolerate more interference and at a very high rate.

The cooperation of technology and signal processing is considered to be the main core to modern communication equipment and grace to them, the objective of digital communication have been met and are still improving with time.

## 1.2 Digital communication system

With the advent of digital integrated circuits and Nano technology, digital communication systems have replaced analogue transmission ones. They offer high data bit rates that meet the needs of the internet, mobile communications and IOT. They also offer data processing options and flexibilities not available with analog transmission.

Digital communication systems are more resistant to the channel noise and they offer the possibility to use technics in error correction. This makes them more reliable during transmission.

Figure 1.1 illustrates the structure of a digital communication system

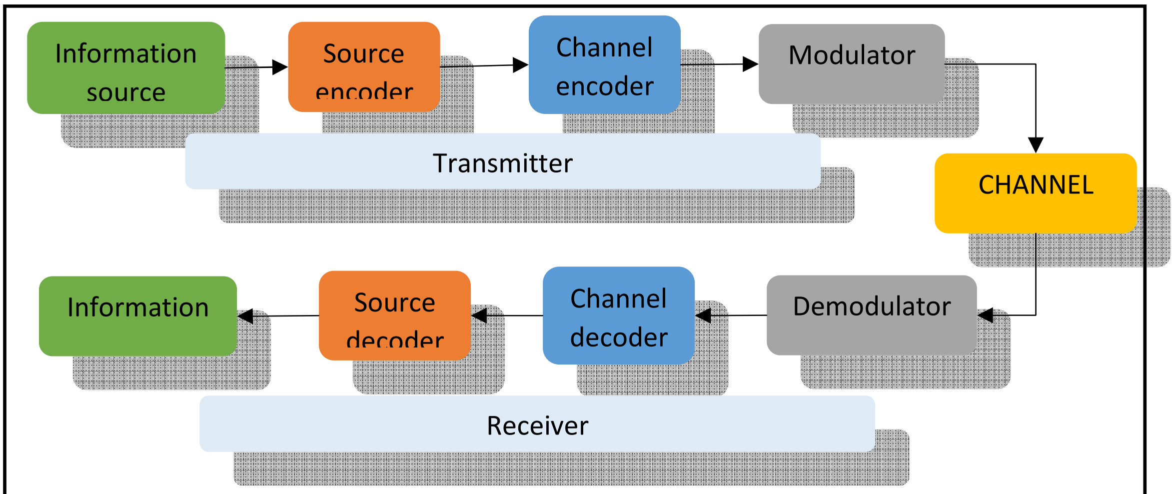



**Figure 1.1 General Structure of Digital Communication Systems**

### 1.2.1 Transmitter

It is composed of

- *Information source* which generate binary sequenced data to be transmitted
- *Source encoder* is designed to minimize redundancies contained in the information bits.
- *Channel encoder* isdesigned to add check bits to the information sequence in order to detect errors and eventually correct them at the receiver.
- *Modulator:* The main objective of modulation is to match the signal to the channel. A modulator maps the encoded digital sequence into a train of short analog waveforms suitable for propagation. Modulation can be performed by either varying the amplitude, phase or the frequency of the sinusoidal carrier wave.
- *Channel* is a transmission media used to carry information. Examples include air (wireless), fiber optic, coaxial cable etc.

### 1.2.2 Receiver

- *Demodulator:* Its input is an analog sequence while its output can either be a Binary or Analog sequence
- *Channel decoder:* It estimates the transmitted message based on the coding rule and characteristics of the channel. Its main objective is to detect and eventually correct the received data corrupted by the effect of the channel noise.

- *Source decoder:* It decodes the received sequence of information by using the coding rule of the transmitter's encoder.

### 1.2.3 Parameters of digital communication

Some of the parameters that characterize a digital communication system are as follows

**a. *Bandwidth* B**

The bandwidth is a measure of how much data can be delivered over a period of time. Signals that change quickly in time have large bandwidth

**b. *Spectral efficiency ψ***

System bandwidth limits the speed at which signals vary; it is quantified by the spectral efficiency $\psi$

$$\psi = \frac{r_b}{B} \text{bits/sec/Hz}$$

(1.1)where $r_b$= data rate B=Bandwidth

**c. *Bit Error Rate* (BER)**

BER is the ratio between the numbers of erroneous received bits to the total number of all the received bits during a time interval.

**d. *Channel capacity***

Channel capacity is the maximum rate at which information can be reliably transmitted over a communication channel.

$$C = B\ log_2\left(1 + \frac{S}{N}\right) \tag{1.2}$$

Shannon's theorem on channel capacity above shows the maximum rate at which information data can be transmitted reliably over a noisy channel.

In reference to Shannon's formula, one can see that noise and bandwidth set a limit to data rates as it can be seen through the calculations of its limits given below.

$$C = Blog_2\left(1 + \frac{S}{N}\right) = \frac{S}{\eta} \times \frac{\eta B}{S} log_2\left(1 + \frac{S}{\eta B}\right) \tag{1.3}$$

$N = \eta B$ Where $\eta/2$ is the two sided noise powerspectral density

$$C = \frac{S}{\eta} log_2 \left(1 + \frac{S}{\eta B}\right)^{\frac{\eta B}{S}} \qquad (1.4)$$

Using the following expression,

$$\lim_{x \to 0} (1 + x)^{\frac{1}{x}} = e \quad (neperian\ base)$$

Equation (1.4) can be re-written as:

If $x = \frac{S}{\eta B}$

$$\lim_{B \to \infty} C = \lim_{x \to 0} C = \frac{S}{\eta} log_2 e = 1.44\ \frac{S}{\eta}$$

## 1.4 Conclusion

In this chapter, different parts of a general communication system have been described. Some of its parameters such as the bandwidth, the channel capacity, the BER and the efficiency are given.

Generally, a good communication system has to operate at low transmission power, tolerate more interference and transmit at high data rate. The design for a channel code has always been a trade-off between energy efficiency and bandwidth efficiency

In 1940s, Shannon came up with a theory known as the Shannon channel capacity. It gives a theoretical upper limit on the data transmission rate for a given bandwidth and the SNR to have an error-free transmission.

Practical codes were unable to operate even close to this theoretical bound. The introduction of concatenated codes, combining simple codes either in series or in parallel resulted to production of codes that achieve a channel capacity nearing the Shannon's theory.

In the following chapter, a turbo encoder, a class of concatenated codes is studied, as it is the main objective of this thesis.

# Chapter 2: Turbo encoder

## 2.1 Introduction

In this chapter, Turbo codes that belong to a class of concatenated code are being described.

## 2.2 Class of concatenated code

Generally, block codes and convolutional codes are used to build concatenated codes. They are classified into two classes

i. Serial concatenated codes
ii. Parallel concatenated codes

### 2.2.1 Serial concatenated codes

A serial concatenated code (SCC) is the one that applies two levels of coding, an inner and an outer code linked together by an interleaver [2]. Each encoder adds redundancy to the output of the previous encoder.

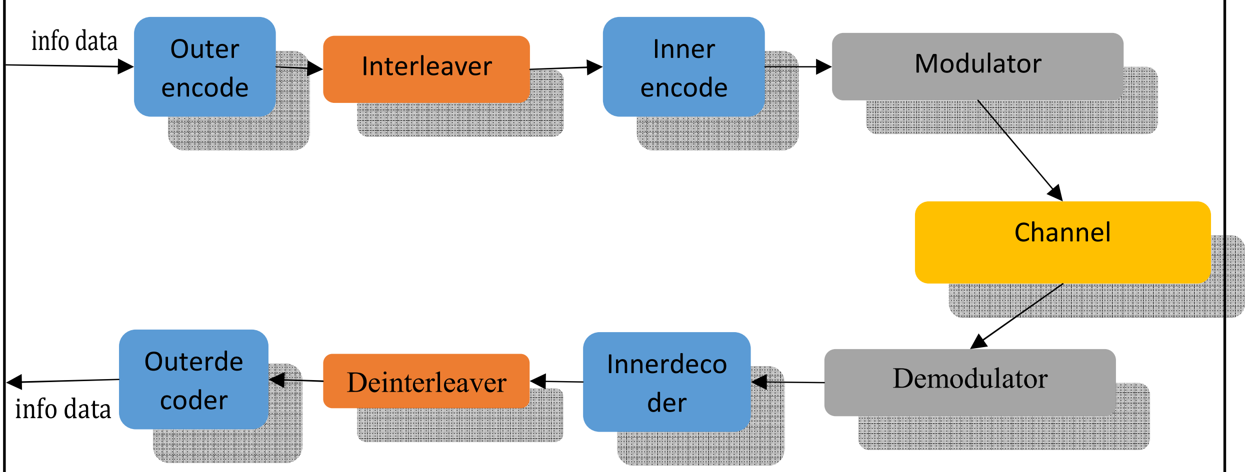



*Figure 2.1: schema of a serial concatenated code [2]*

*Considering* an outer code $C_o$ with rate $R_o = \frac{k}{p}$ and an inner code $C_i$ with rate $R_i = \frac{p}{n}$ . Then the overall concatenated code rate R becomes [2]

$$R= R_i x\ R_o = \frac{k}{p} \times \frac{p}{n} = \frac{k}{n} \qquad (2.1)$$

When decoding SCCs, the codes are decoded in reverse order, to how they were encoded in the encoder.

### 2.2.2 Parallel Concatenated codes

.Parallel-concatenated codes (PCCs) are also known as **turbo codes** [3]. Two or more recursive systematic encoders separated by an interleaver form them. PCCs encoders encodes the same input bits but are arranged in a different manner, thanks to the interleaver

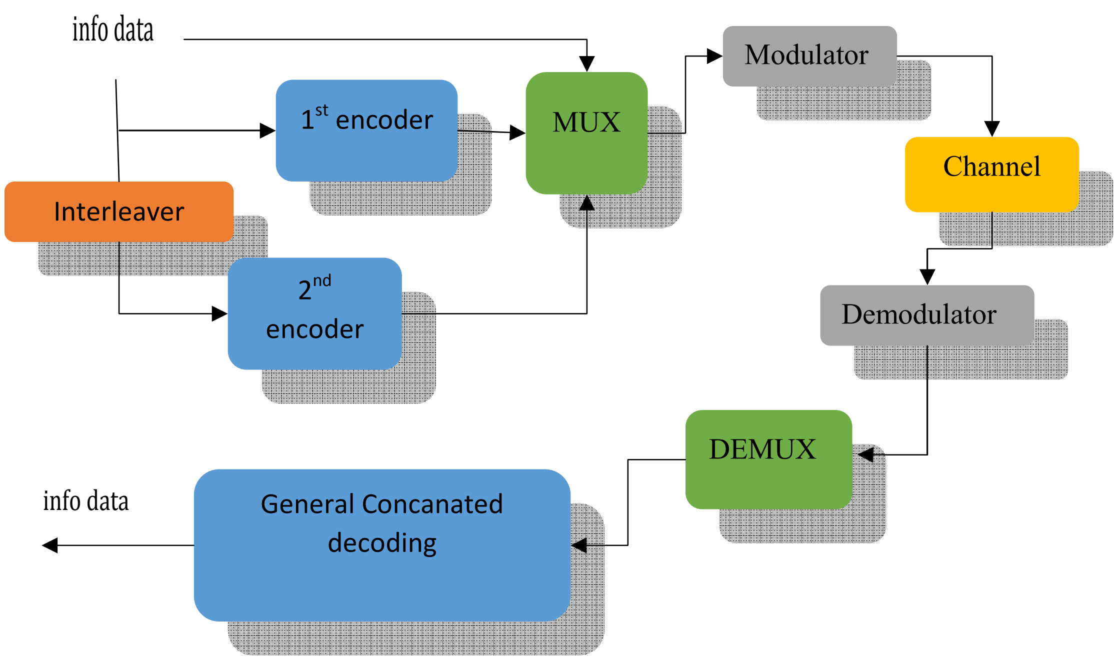



*Figure 2.2: schema of a parallel concatenated code*

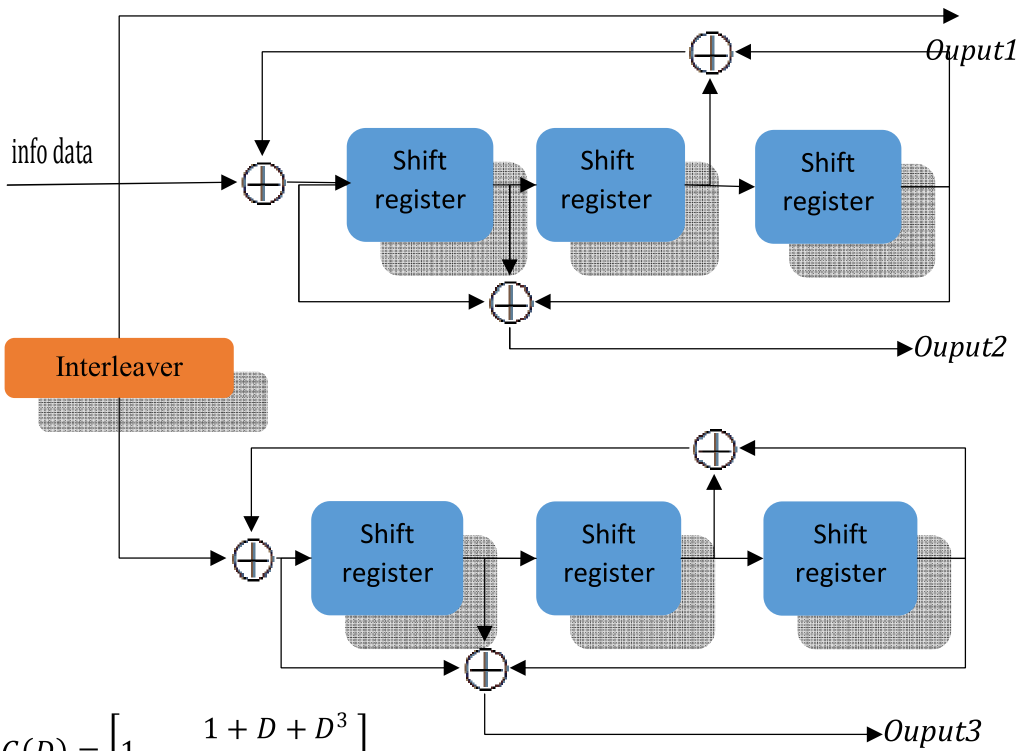
The global encoding rate for parallel-concatenated codes is given by [2]:

$$\frac{1}{R_{Total}} = \frac{1}{R_1} + \frac{1}{R_2} \tag{2.2}$$

Turbo codes and serial concatenated codes differ in so many ways, among them is the global rate in PCCs is higher than the global rate of a serial concatenated code.

## 2.3 Convolutional turbo codes

A convolutional turbo code is the parallel concatenation of two or more identical recursive systematic convolutional (RSC) encoders with a large memory and reasonable complex decoding operations. Figure 2.3 shows an example of a convolutional turbo code.




$$G(D) = \left[ 1 \quad \frac{1 + D + D^3}{1 + D^2 + D^3} \right]$$

Figure 2.3 a convolutional turbo code

In order to understand convolutional turbo codes, there is need to know about convolutional codes.

In the following sections, convolutional codes and the interleaver are discussed in details

## I. Convolution codes

Peter Elias first introduced convolution codes in the 1955 though Forney performed its groundwork. Convolutional codes are widely used in space communications, cellular mobile, digital video broadcasting etc.

Due to its simple structure and the availability of easily implemented maximum likelihood soft decision decoding technics, convolutional codes became one of the most popular codes known. [2].

### a. Structure of convolutional codes

A convolutional code $C(n, k, m)$ is generally described by its parameters

k = input data

n = output data

m = memory of the convolutional encoder

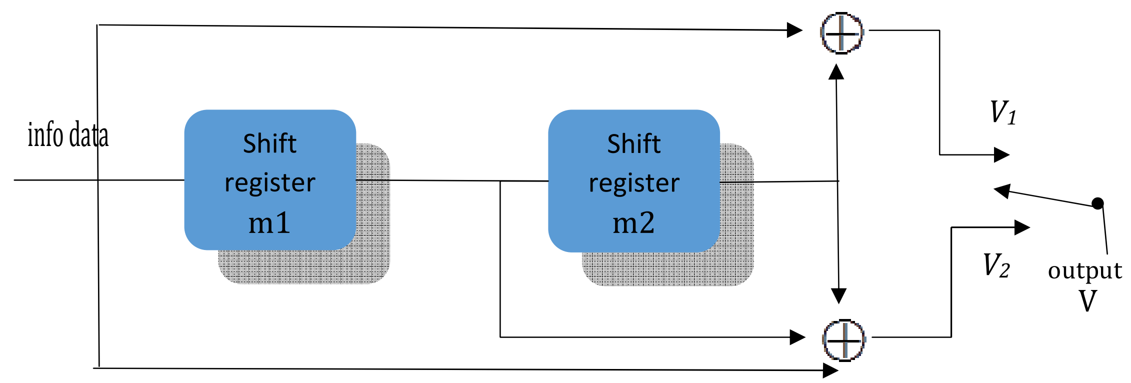



Figure 2.4 C(2, 1, 2) convolutional encoder

Figure 2.4 shows a C(2, 1, 2) convolutional encoder

The elements of the output code are calculated as follows [4]

$V_1 = U_0 \oplus U_2$

$$V_2 = U_0 \oplus U_1 \oplus U_2 \tag{2.3}$$

The addition sign ⊕ represents the modulo-2 addition.

Since the encoder has a memory of 2, then the encoder is said to have a constraint length of two.

The code is specified by a set of generator sequence of length (m+1).

The interconnections between the shift register and the modulo-2 adders (lines connecting them) form the generator sequence.

Considering the generator of *figure 2.4,*

$g^1 = (g_0^1, g_1^1, g_2^1) = 1\ 0\ 1$ $\begin{cases} 1, & participating \\ 0, & not\ participating \end{cases}$ $g^2 =$

$(g_0^2, g_1^2, g_2^2) = 1\ 1\ 1$ (2.6)

It specifies which inputs are participating in code generation.

$V_i = U * g^i$

$$V_i = U_0 g^i \oplus U_1 g^i \oplus U_2 g^i \oplus \ldots U_k g^i \qquad (2.7)$$

### b. Polynomial Representation of Convolutional codes

The inputs, outputs, and generators can be represented by a power series sequence. This is done by introducing a symbol D, a unit of delay operator. [3]

The power denotes the number of times the binary symbol is delayed with respect to the initial binary input symbol in the sequence.

$D^0 = 1$ *0 delay*

$D^1 = D$ *memory delay of 1*

$D^2 = 1$ *memory delay of 2*

The co-efficient of the polynomials are the shift register taps

$$V_i(D) = U(D) g^i(D) \qquad (2.8)$$

Example

$g^1 = (1\ 0\ 1) \Longrightarrow g^1(D) = 1 + D^2$

$g^2 = (1\ 1\ 1) \Longrightarrow g^2(D) = 1 + D + D^2$

$u = (1\ 0\ 1\ 1\ 1\ 0\ 0) \Longrightarrow U(D) = 1 + D^2 + D^3 + D^4$

$$G(D) = [1 + D^2 \quad 1 + D^2 + D^3 + D^4]$$

$$V_1(D) = U(D) g^1(D)$$

$$V_2(D) = U(D) g^2(D) \qquad (2.9)$$

### c. State diagram

Since a convolutional encoder is a sequential circuit (synchronized), a state diagram can describe it. A state of the encoder is defined by its memory content (m).

There are $2^m$ number of possible states for a (n, 1, m) convolutional encoder.

The previous state and the current input determine current state and the output of the decoder. The encoder changes its state when a message block is shifted into the encoder.

Using figure 2.4 as our convolutional encoder, its state diagram given in figure 2.5 is constructed from table 2.1 below, which gives the output bits for each input bit, the initial and the next state of the encoder [4].

| Input | Initial state | Next state | output |
|---|---|---|---|
| 0 | 00 | 00 | 00 |
| 1 | 00 | 10 | 11 |
| 0 | 01 | 00 | 11 |
| 1 | 01 | 10 | 00 |
| 0 | 10 | 01 | 01 |
| 1 | 10 | 11 | 10 |
| 0 | 11 | 01 | 10 |
| 1 | 11 | 11 | 01 |

Table 2.1 input and output states of the encoder

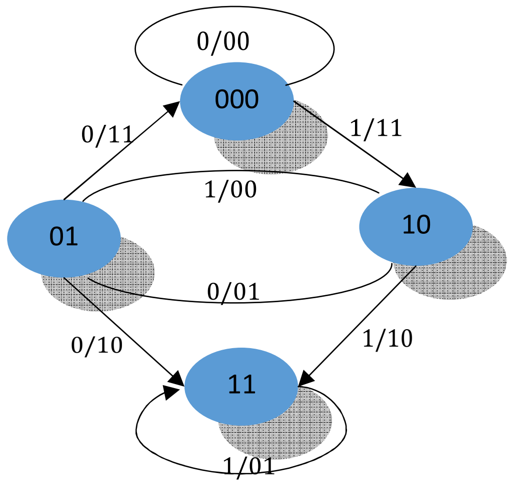



figure 2.5 a state diagram

### d. Trellis diagram

A trellis diagram is the most common representation of a convolutional code. It is of major importance for defining the properties of a code and for decoding.

A trellis diagram is an upgrade of a state diagram obtained by expanding in time every state transition and every path starting from the all zero state [2]. The output code sequence can be obtained in the trellis by tracing the path specified by the input sequence. The trellis diagram

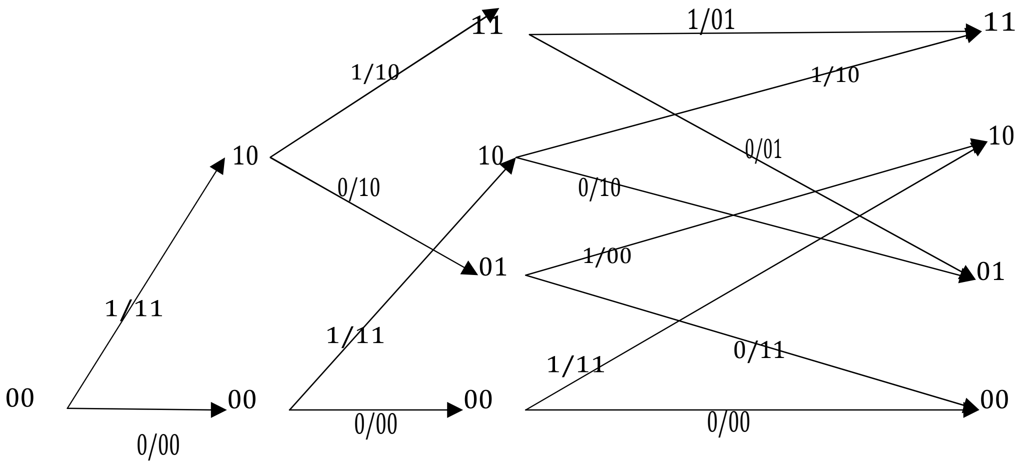



Figure 2.6 Trellis diagram of a (2.1.2) convolutional encoder

of a C (2.1.2) encoder is shown in figure 2.6

### e. Classification of convolutional codes

Convolutional codes can be classified based on

- Input and output connections
  - Feed forward encoder (Non-recursive encoder)
  - Feedback encoder (Recursive encoder)
- Systematic encoders
- Non-systematic encoders
- Equivalent encoder

### i. Feed forward encoder

A feed forward encoder is also referred to as a non-recursive encoder [5]. Its generator matrix

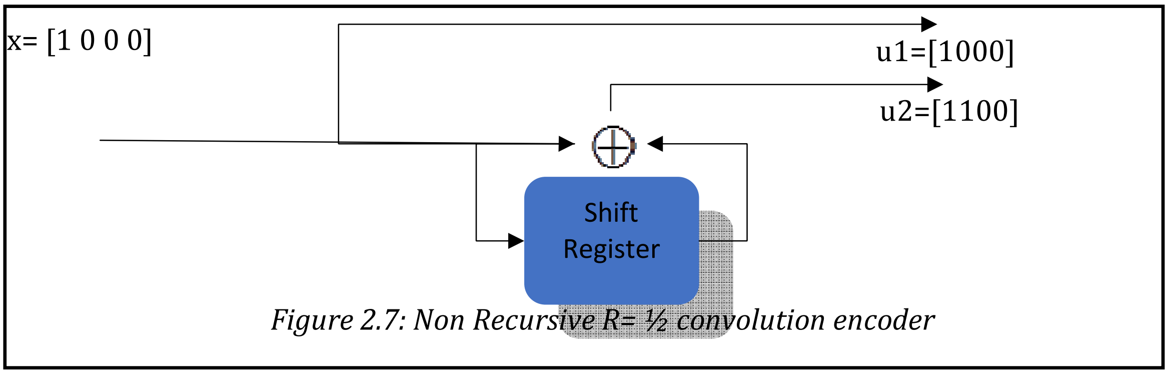



*Figure 2.7: Non Recursive R= ½ convolution encoder*

does not contain a feedback path in an encoder.

### ii. Feedback encoder

A feedback encoder also known as a recursive encoder is an encoder obtained by feeding back

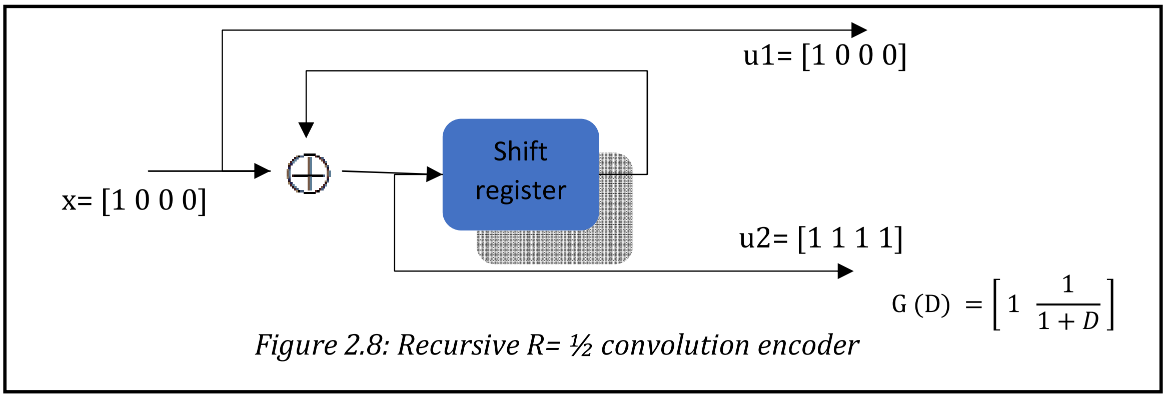



*Figure 2.8: Recursive R= ½ convolution encoder*

one of its encoded outputs to its input.

*<u>Remark</u>*

It can be observed that a recursive convolutional encoder produces code words with increased weight compared to a non-recursive encoder, this leads to better error performance when RSC are used during coding.

### iii. Systematic encoder

A systematic encoder is an encoder whose coded message contains the message to be transmitted, to which redundant information is added.

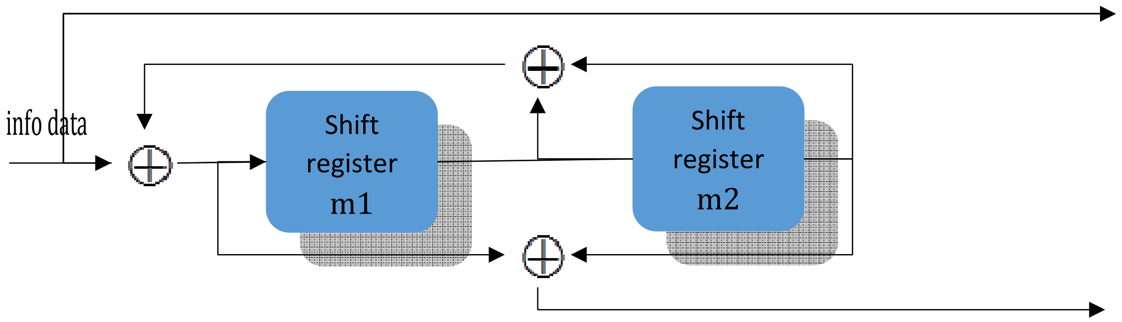



figure 2.9 A systematic encoder

The figure 2.9 shows an example of a systematic encoder

Its generator matrix is of the form. [2]

$$G(D) = \left| I \quad P(D) \right| \tag{2.10}$$

Where *I= k*k identity matrix*

*P (D) = k * (n-k) matrix*

### iv. Non-systematic encoder

A non-systematic encoder is an encoder whose coded word is modulo-2 calculated output using the information data and the generator polynomial of the encoder [6].

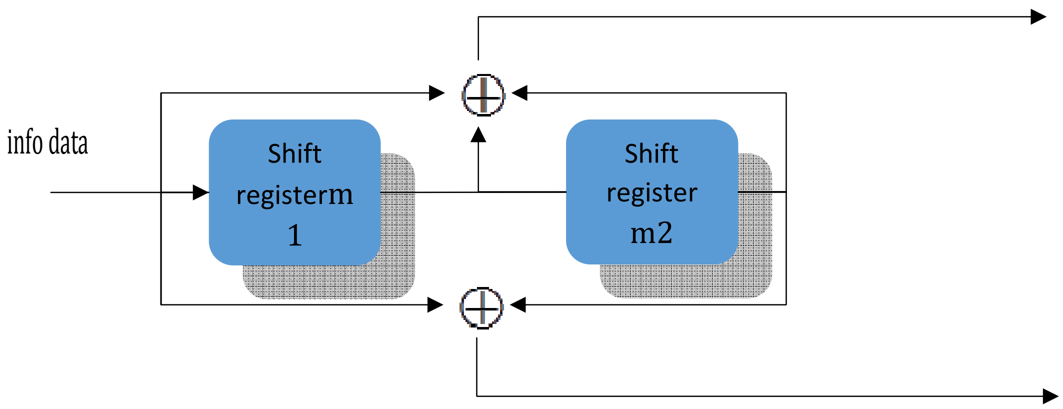



figure 2.10 A non-systematic encoder

### v. Equivalent encoder

Two convolutional encoder are said to be equivalent if their generator matrices G (D) and G' (D) are equivalent. This means that they encode the same code word. [7]

Two generator matrices G (D) and G' (D) are equivalent if and only if there exists a rational invertible matrix T (D) such that

$$G'(D) = T(D) * G(D)$$

*If*

$$V(D) = U(D) * G'(D)$$

$$V(D) = U(D)\, T(D) G(D)$$

$$V(D) = U'(D) * G(D) \tag{2.11}$$

The input sequences that generate the same codeword are different and may have different weights.

An equivalent encoder is used to obtain a systematic form for a given generator matrix of a convolutional encoder.

#### f. Conversion from non-systematic to systematic encoders

For systematic encoders, the message can be read out directly from the received sequence. On the other hand, with non-systematic encoders, the message is not visible.

A recursive systematic convolutional (RSC) encoder is obtained from the non-recursive nonsystematic convolutional encoder by feeding back one of its encoded outputs to its input.

To convert from a non-systematic to a systematic encoder, the following steps are done;

$G(D) = [\,g_1 \quad g_2]$ Non-systematic

$G'(D) = [I \quad P(D)] = \left[1 \quad \frac{g_2}{g_1}\right]$ Systematic

$I$= identity matrix (2.12)

If $G(D) = [1 + D + D^2 \quad 1 + D^2]$

$G'(D) = T(D) * G(D)$

$T(D) = \frac{1}{1+D+D^2}$

$G'(D) = \frac{1}{1+D+D^2} * [1 + D + D^2 \quad 1 + D^2]$

$G'(D) = \left[1 \quad \frac{1+D^2}{1+D+D^2}\right]$

In order to recover the information from the decoded sequence of a non-systematic encoder theoretically, it is passed through an inverter specified by the generator matrix $G^{-1}$ (D).

$V(D)\, G^{-1}(D) = U(D) G(D)\, G^{-1}(D)$

*Condition*

$$G(D)\, G^{-1}(D) \quad (2.13)$$

If the inverse generator matrix $G^{-1}$ (D) does not exist, the code is said to be catastrophic.

### g. Punctured convolutional codes

A code rate of a convolution code can be varied without changing the trellis structure. This is done by not transmitting certain code symbols or puncturing the original code sequence. [2]

The code has a simpler trellis structure compared to the original ½ code (*figure 1.4)* though its encoding and decoding operations are complex.

The figure 2.11 below shows the trellis diagram of a punctured convolutional code where x represents the punctured (removed) bit. Its generator polynomial is given by

$$G(D) = \begin{vmatrix} 1+D & 1+D & 1 \\ 0 & D & 1+D \end{vmatrix} \quad (2.14)$$

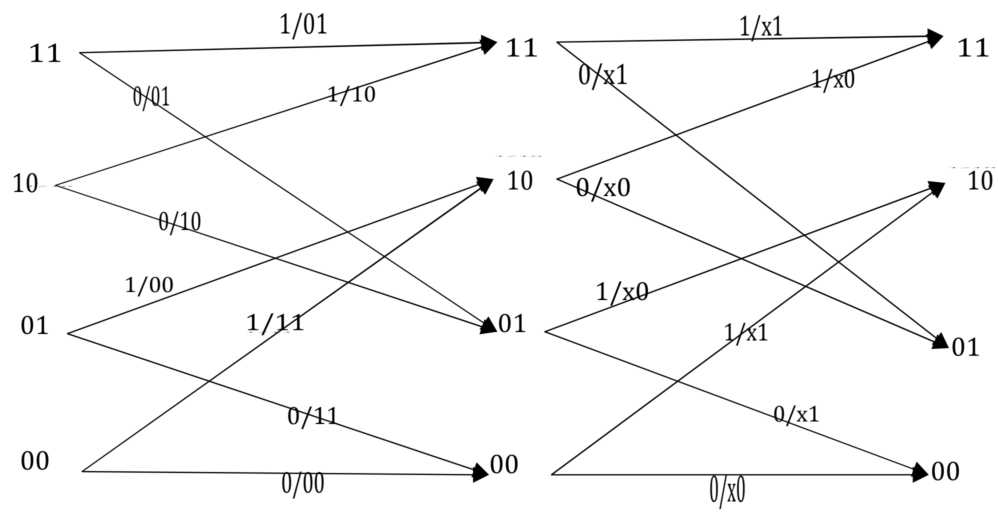



Figure 2.11 Trellis diagram of a punctured convolutional encoder

Punctured codes have an advantage particularly for high rate applications.

## 2.4 Interleaving

In a turbo encoder, an interleaving function is Inserted in between the encoders to rearrange the information bits in a pseudo random known manner before being coded by the second encoder.

### 2.4.1 Interleaving technics

The use of interleaver plays a fundamental role in turbo coding schemes therefore its design and performance analysis has to be effective.

There are many types of interleaver based on their design, among them among them that are frequently used include [2, 5]:

- Block type interleaver
- Convolutional interleaver
- Random interleaver

## a. Block type interleaver

A type of block interleaver is shown in figure 2.3. The input data sequence is organized into a matrix, which is obtained dividing the input sequence into say m segments and arranged into m rows. The number of columns n depend on the size of the segment. It is later on read out as columns.

The format of the input sequence is in form of a matrix N

*N=m x n* *m* rows and *n* columns.

Example

If the input sequence is: [$a_0$, $a_1$, $a_2$, … $a_{11}$]

$$\begin{bmatrix} a_0 & a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 & a_7 \\ a_8 & a_9 & a_{10} & a_{11} \end{bmatrix}$$

Figure 2.12 A block interleaver

If it is divided into four segments, it can be arranged into the following matrix

The interleaved serial data [$a_0$, $a_4$, $a_8$, $a_1$, $a_5$, $a_9$, $a_2$, $a_6$, $a_{10}$, $a_3$, $a_7$ $a_{11}$] read out as column is fed into the second encoder

## b. Convolutional interleaver

Ramsey and Forney proposed a convolutional interleaver as shown in figure 2.4. It consists of an input, an output switches and a bank of m shift registers (memory). The information sequence is arranged in blocks of bits. The input switch cyclically inserts each block of bits into the bank of the registers; the output switch cyclically samples the bank of registers in the same order as the input switch. [2]

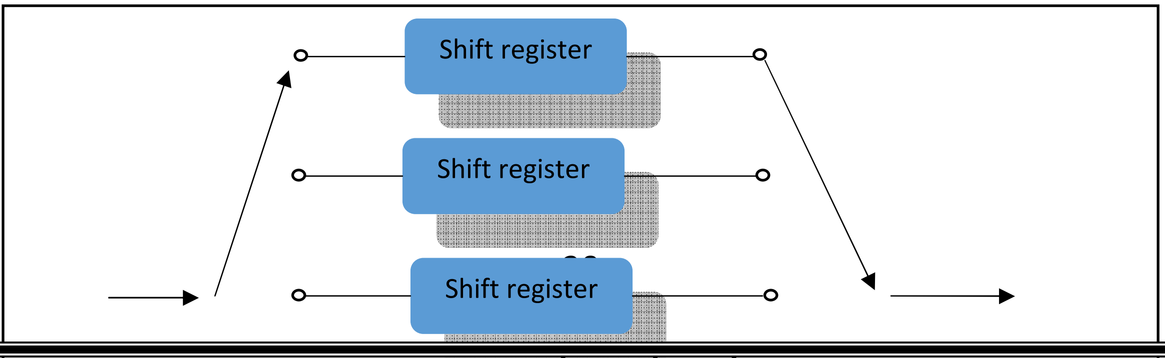



Figure 2.13 A convolutional interleaver

### c. Random interleaver

In such an interleaver, the information sequence is arranged in blocks of N input bits and are read into and out of the interleaver in a random manner. [5]

Figure 2.5 shows inputs and outputs of a random interleaver

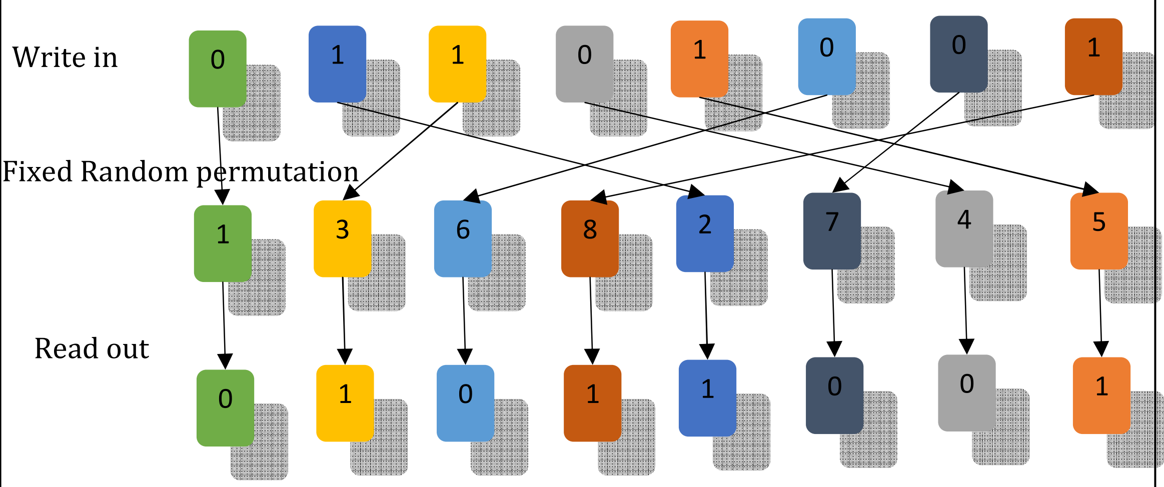



figure 2.14 A random interleaver

The random interleaver receives [0 1 1 0 1 0 0 1] as the input and reads out [0 1 0 1 1 0 0 1] as its output.

## 2.5 Conclusion

In this chapter the general principles behind serial concatenation and parallel concatenation also referred to as Turbo codes have been discussed.

The main components of a convolutional turbo code encoder are Recursive Systematic Convolutional (RSC) encoders separated by an interleaver that link them. The choice of the interleaver plays an essential part in the performance of the code.

# Chapter 3: Turbo decoding

## 3.1 Introduction

Earlier decoding, decoders would make hard decision on the received sequence to get a discrete value of the transmitted data. The disadvantage of such an approach was some bits were determined with greater certainty than others were [8].

A better rule was implemented known as maximum likelihood estimation (MLE) due to the disadvantages of hard decision encoding. It puts into account the priori probability of the inputs, whereby if the probability of a symbol is above the threshold probability, the symbol is +1, if it falls below, the symbol is 0. [2]

The maximum posteriori (MAP) is a special type of MLE that takes into consideration the conditional probability [2]

Another algorithm used for turbo decoding is a trellis based decoding algorithm, its output is in form of hard-quantized estimation of the symbols. It was later modified to generate soft outputs instead of hard and became known as soft output Viterbi algorithm (SOVA). [8]

## 3.2 Turbo decoding

Figure3.1 below shows a general system model of a decoder

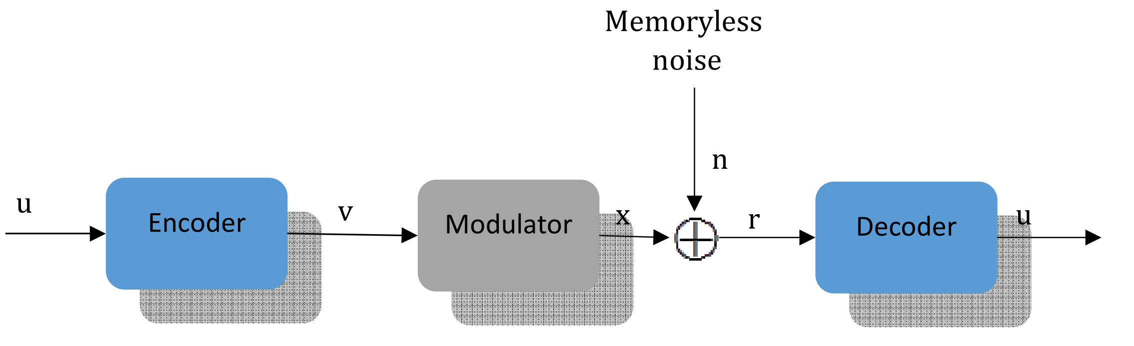



*Figure 3.1: A general system model of a decoder*

A binary sequence denoted by u is generated and encoded by the encoder.

$$u_k = (u_0, u_1, u_2, u_3, u_4 \ldots u_k) \tag{3.1}$$

The modulator modulates the output v of an encoder, which depends on the type of encoder used. Its output is denoted by x.

$$v_k = (v_0, v_1, v_2, v_3, v_4 \ldots v_k)$$

$$x_k = (x_0, x_1, x_2, x_3, x_4 \ldots x_k) \tag{3.2}$$

The modulated sequence is transmitted through the channel where it combines with noise and later on received as $r_i$

$$r_i = (r_0, r_1, r_2, r_3, r_4 \ldots r_i)$$

$$r_i = x_i + n_i \tag{3.3}$$

$n_i$ can be a zero-mean random Gaussian noise variable.

Each noise sample is assumed independent from each other. The decoder decodes the received data then gives an estimate of the information sent by processing the received sequence.

### 3.2.1 Turbo decoder

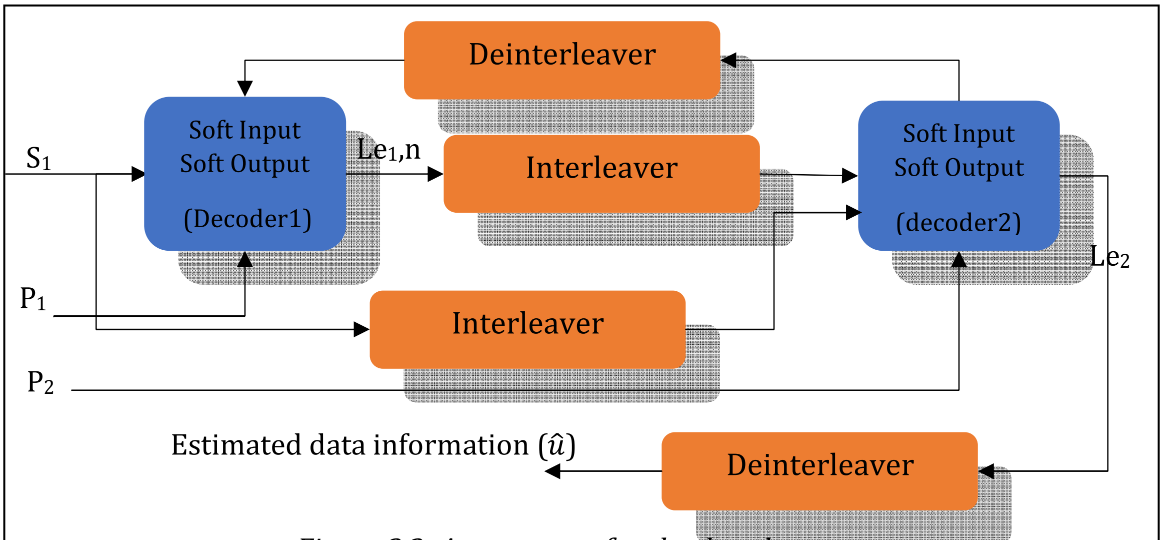



*Figure 3.2: A structure of turbo decoder*

Figure 3.2 below shows a general structure of a turbo decoder [3].

The systematic received data ($S_1$) and the first parity data ($P_1$) enter the first decoder. Its output, the extrinsic information on the message ($Le_1$ (soft output)), is then interleaved before being fed to the second decoder together with the other parity information ($P_2$). The second decoder also produces a soft output $Le_2$, a newer guess of the message.

For a new iteration n to begin, the newer guess of the information is de-interleaved and delivered to the first decoder.

After a certain number of iterations, the log-likelihood ratio (LLR) at the output of the second decoder is de-interleaved and undergoes hard decision, to give the final estimate of the information data$\hat{u}$.

In the turbo decoder, decoding process depends on a couple of algorithms, some of them include:

- The MAP algorithm
- The LOG-MAP algorithm
- The Max LOG-MAP algorithm

In the following section, a brief description of the algorithms are given. The details of different calculations are given in the annex A.

### a. The MAP algorithm (BCJR) [3, 9]

The MAP algorithm is a special type of MLE rule. For each transmitted symbol, a soft output is generated in the form of *a posteriori* probability (APP)$\boldsymbol{P_r}\,(\boldsymbol{u}/\boldsymbol{r})$based on the received sequence r.

The APP values are computed in the logarithmic form of likelihood ratio $L\ (u_r/r)$ known as the log likelihood ratio (*LLR*) [3], which is given by:

$$L(u_r/r) = ln\frac{P\ (u_r=+1/r)}{P\ (u_r=-1/r)} \qquad (3.4)$$

For short constraint length of convolutional codes, the equation is simplified using the trellis structure of the code. [3]

Equation (3.4) can be formulated as:

$$L(u_r/r) = \ln\frac{\sum_{(+1)} P(s',s,r)}{\sum_{(-1)} P(s',s,r)} \qquad (3.5)$$

P(s’, s, r) represents the joint probability of receiving the bit sequence r being in state (s′) and state (s) that correspond to either positive or negative values on the trellis diagram.

$$P\ (u_r=+1/r) = \frac{\sum_{(+1)} P(s',s,r)}{P(r)} \qquad (3.6)$$

$s'$Represents the initial state $s$ Represents the next state

It can be further explained using [9]

- Branch state metric γ
- Forward state metric α
- Backward state metric β

Figure 3.3 shows the three branches in a trellis diagram.

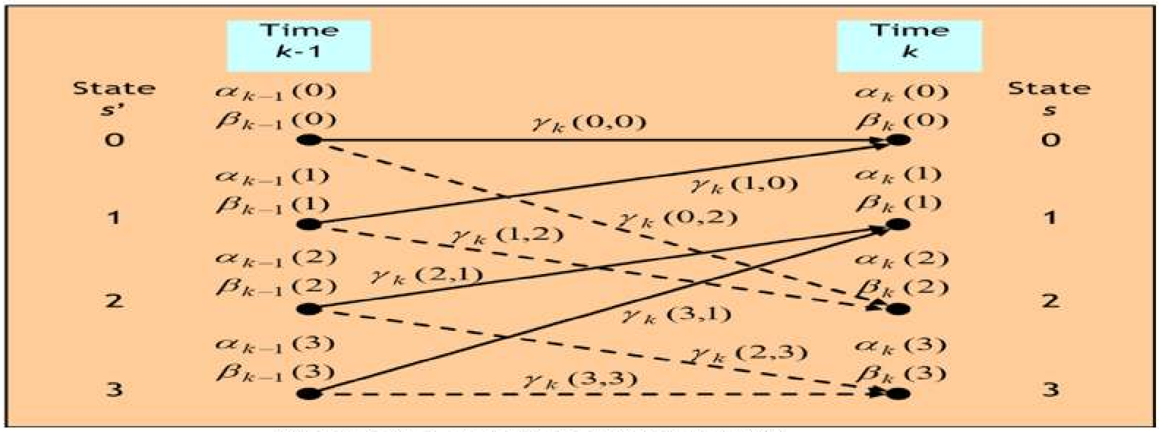



*Figure3.3: γ, α and β as trellis labels [9]*

- *Forward state metric (α-Alpha)*

The equation represents the forward state metric

$$\alpha_k(s) = \sum_{s'} \gamma_k(s', s)\alpha_{k-1}(s') \quad (3.7)$$

- *Backward state metric (β- beta)*

The equation represents the forward state metric

$$\beta_{k-1}(s') = \sum_{s} \gamma_k(s', s)\beta_k(s) \quad (3.8)$$

- *Branch metric(γ-gamma)*

Depends on the channel

$$\gamma(s', s) = P(u_k)P(r_k/v_k) \quad (3.9)$$

γ For AWGN is the equation can be written as [9]

$$\gamma(s', s) = P(u_k)e^{u_k\frac{L(u_k)}{2}} e^{-\frac{Es}{No}2ar_kv_k} \quad (3.10)$$

$\boldsymbol{P(u_k)}$represents the input a prior probability ratio, it will cancel out when computing the LLRs**a** is the amplitude fading factor (a=1 for a non-fading channel)

The final output of the decoder (log like ratio) therefore becomes [9].

$$\mathrm{L}(u_r/r) = \ln \frac{\sum_1^+ P(s', s, r)}{\sum_1^- P(s', s, r)}$$

$$P(s', s, r) = \alpha_{k-1}(s')\gamma_k(s', s)\beta_k(s)$$

$$\mathrm{L}(u_r/r) = \ln \frac{\sum_1^+ \alpha_{k-1}(s\prime)\gamma_k(s', s)\beta_k(s)}{\sum_1^- \alpha_{k-1}(s\prime)\gamma_k(s', s)\beta_k(s)} \qquad (3.11)$$

*Remark*

Implementing the MAP algorithm is very complex due to the multiplication and division operations. However, implementing it in the logarithmic domain, the complexity is greatly reduced [3, 9].

### b. Log MAP algorithm

The logarithm version of the MAP algorithm (the log MAP algorithm) reduces the complexity of calculations. The logarithm and exponential computations can be eliminated by the following approximation [3, 9].

$$\mathrm{Max}^*(x, y) = \ln (e^x + e^y)$$

$$= x + \ln (1 + e^{-|x-y|})$$

$$= \max^* (x, y) + \ln (1 + e^{-|x-y|}) \qquad (3.12)$$

i. *The branch metrics can be written as*

$$\gamma^*(s', s) = \ln \gamma(s', s) = \ln P(u_k) + \frac{u_k\, L(u_k)}{2} + \frac{L_c}{2} r_k v_k \qquad (3.13)\; L_c = 4a\frac{E_s}{N_0}$$

ii. *Forward metrics can be written as*

$$\alpha_k^*(s) = \ln \alpha_k(s) = \max^*[\gamma_k^*(s', s) + \alpha_{k-1}(s')] \qquad (3.14)$$

iii. *Backward metrics*

$$\beta_{k-1}^{*}(s')) = \ln\ \beta_k = max^{*}[\gamma_k^{*}(s',s) + \beta_k(s)] \tag{3.15}$$

*The APP value*

$L(u_r/r) = max^{*}(s',s) \in$

$\sum_l^{+}[\alpha_{k-1}{}^{*}(s') + \gamma_k{}^{*}(s',s) + \beta_k{}^{*}(s)] -$ $max^{*}(s',s) \in \sum_l^{-}[\alpha_{k-1}{}^{*}(s') +$

*γk∗s′,s+βk∗(s)*

(3.16)

The addition and subtraction sign have replaced the multiplication and division sign.

### c. Max log MAP algorithm

Due to the decoding delay and complexity of hardware in Log MAP algorithm, it was modified to become Max Log MAP. The correction function in the Log MAP equation fc is considered as a negligible term in Max-Log-MAP algorithm. [10]

$$fc = \ln(1 + e^{|y-x|}) \tag{3.17}$$

$$\max{}^{*}(x, y) \approx \max(x, y) \tag{3.18}$$

Modification to Max Log MAP comes at the expense of some performance degradation.The max-log MAP algorithm is the least complex though it offers the worst BER performance. From the hardware point of view, the max algorithm is less complex. [10]

## 3.3 Iterative decoding

Iterative decoding shown in figure 3.4 below consists of two component decoders serially connected by an interleaver identical to the one used in the encoder [3, 9].

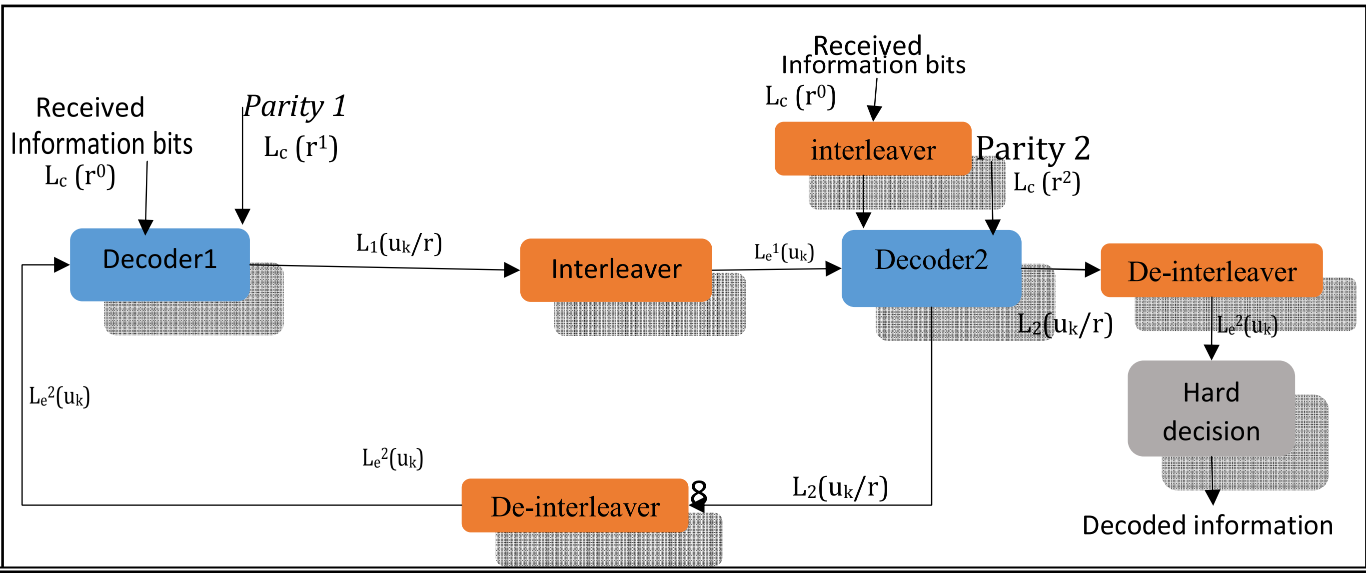



Figure 3.4 Schema of turbo iterative decoding

Iterative Turbo Decoding achieves a good performance as the number of the iterations increases along with the decoding process. However too many iterations cause computational burden and latency.

For iterative decoding, it is possible to use the SOVA, MAP, LOG-MAP and MAX LOG-MAP algorithms [9].

For an iterative turbo decoder of rate 1/3, two decoders for each RSC encoder are used. Three outputs are received from the channel r= ($r^0$, $r^1$, $r^2$) which correspond to information bits, and the two parity bits.

$$r = (r^0_0\, r^1_0\, r^2_0,\quad r^0_1\, r^1_1\, r^2_1,\quad r^0_2, r^1_2, r^2_2 \ldots)$$

For the AWGN channel, we define the log like ratio of the transmitted bit given the received bit as [3, 9]

$$L\,(u_k/r^0_k) = 4a\frac{Es}{No}r^0 + \ln\frac{P(u_k=+1)}{P(u_k=-1)}$$

$$L\,(u_k/r^0_k) = L_c\,(r^0) + L\,(u_k) + L_e\,(u_k) \qquad (3.19)$$

Where $\frac{Es}{No}$ is the channel SNR,

Lc $=4a\frac{Es}{No}$ is the channel reliability factor where a is the amplitude fading factor (a=1 for a non-fading channel)

$\ln\frac{P(u_k=+1)}{P(u_k=-1)}$ The prior knowledge about the information,

$L_e\,(u_k)$ is the extrinsic information

**For the first iteration, $L_e\,(u_k)$ =0,**

The decoder1 produces the extrinsic information L ($u_k$/r) on the message when it receives soft L-values ($L_c$ ($r^0$) for information bit and $L_c$ ($r^1$) for parity 1 bit) from the channel.

The extrinsic information contains two terms

$$L_e{}^1 = \mathrm{L}\,(u_k/r) - [L_c(r^0) + L(u_k)] \qquad (3.20)$$

Subtracting the terms removes the effect of the information bit $u_l$ making it an independent estimate of the information

The extrinsic information L ($u_k$/r) undergoes appropriate interleaving before again being fed to decoder 2 as $L_e^1$ ($u_k$) together with the received soft channel L-values of the other parity information $L_c$ ($r^2$)

- In the initial iteration of decoder 1, the extrinsic L-values is 0. They are then passed from one decoder to another during the iterative decoding process.
- After sufficient number of iterations, the LLR at the output of decoder 2 $L_e^2(u_k)$ is de-interleaved and delivered for hard decision, which estimates the information bit based only on the sign of the de-interleaved LLR [9].

## 3.4 Conclusion

In this chapter, the principles of turbo decoding, the algorithms used which calculate a posteriori probabilities (APPs) and the concept of iterative decoding which depends on the use of SISO (Soft Input Soft Output) decoders are discussed. An optimal Algorithm for computing APPs is the MAP (or BCJR) algorithm. The others are just an improvement to it.

Standard turbo decoding uses fixed number of iterations. At the time of termination, the data block is either successfully decoded or it cannot be decoded at all.

# Chapter 4: Implementation and analysis of turbo codes

## 4.1 Introduction

This chapter presents implementation and analysis of LTE and WiMAX Turbo codes using Matlab Software. Different parameters such as varying number of decoding iterations and frame length were used to verify their performance.

Observing the effect of Additive White Gaussian Noise and the Rayleigh channel, different decoding algorithm processes (log MAP and Max log MAP) were used to analyze the Code performances under such conditions.

The performances of the turbo code were studied in this work by analyzing

a. *The Bit Error Rate (BER)*
b. *Energy per bit to noise power spectral density ratio (Eb/No)*
c. *The execution time*

## 4.2 Parameters used in the simulation

Table 4.1 given below shows the elements used in the simulation

| | |
|---|---|
| Input signal | Random signal and an image |
| Type of modulation | BPSK |
| Noise | Random AWGN |
| Canal | Rayleigh fading channel |
| Interleaver | Random |
| Turbo code | LTE Turbo Code<br>WiMax Turbo Code |

Table 4.1 turbo code program elements

Figure 4.1 shows a flow chart of the program used during simulation.

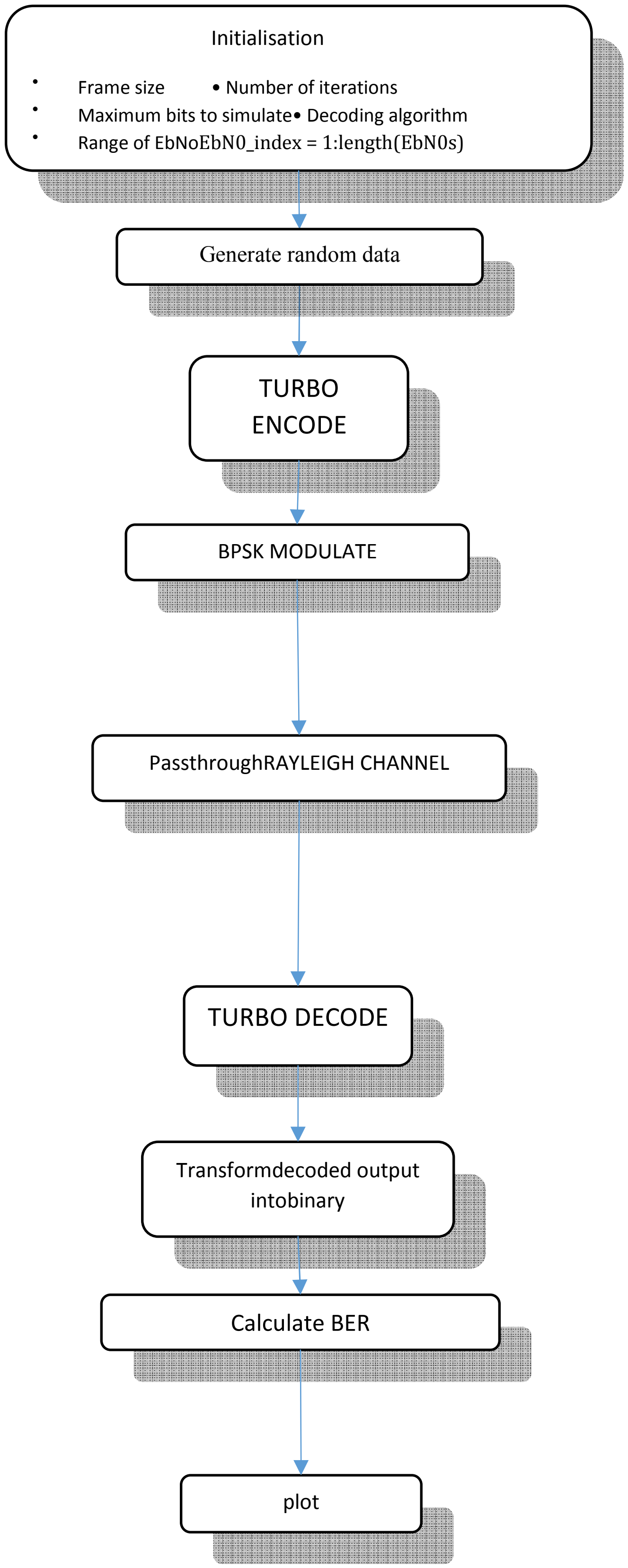



*Figure4.1: Control flow chart of a Turbo code MATLAB Implementation*

yes

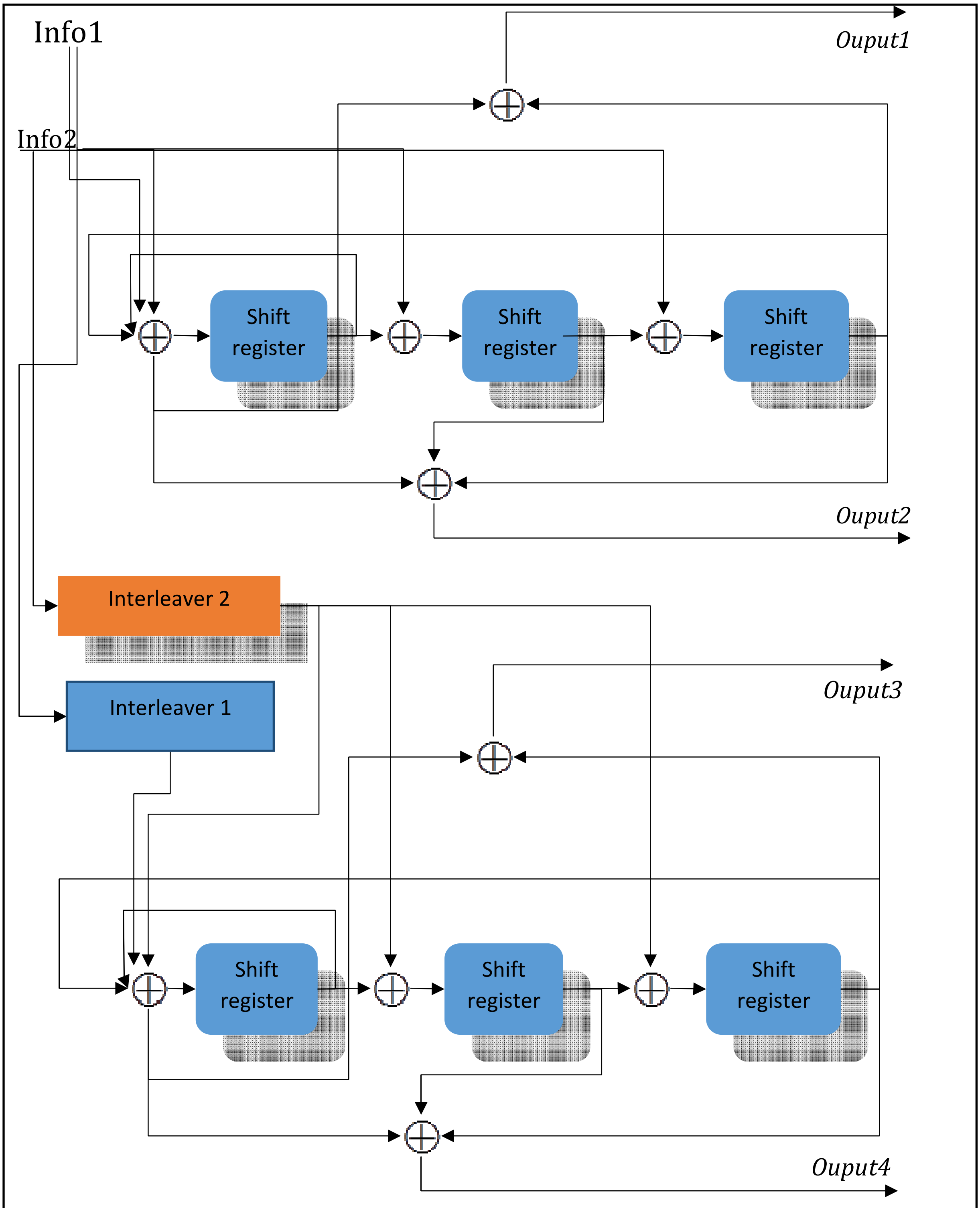
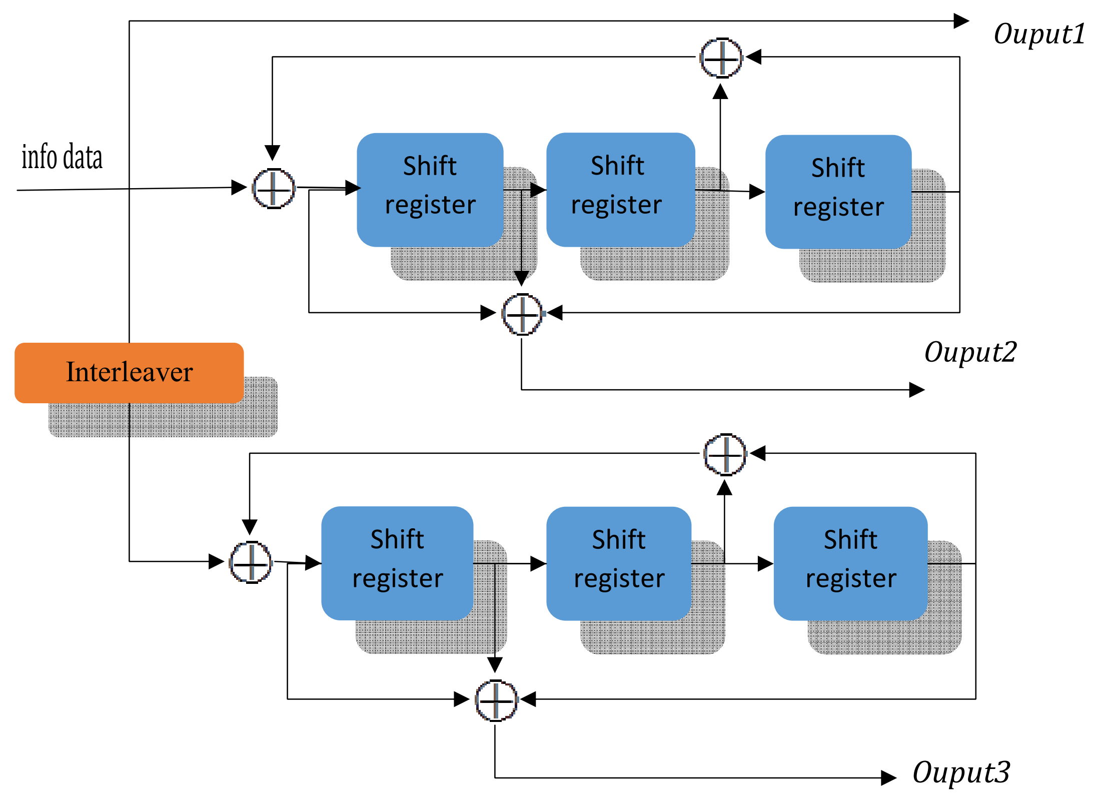



*Figure4.2: WiMAX turbo encoder*

Figure 4.2 represents WiMAX turbo encoder used during the simulation

Figure 4.3 shows the LTE turbo encoder used during the simulation




$$G(D) = \left[1 \qquad \frac{1 + D + D^3}{1 + D^2 + D^3}\right]$$

*Figure4.3: LTE turbo encoder*

Table 4.2 given below gives turbo encoder parameters used in the simulation

| | LTE Turbo Code | WiMAX Turbo Code |
|---|---|---|
| Rate | 1/3 | 1/2 |
| No of memories | 3 | 3 |
| No of states | $2^3$=8 | $2^3$=8 |
| No of inputs | 1 | 2 |
| No of outputs | 3 | 4 |

Table 4.2 turbo encoder parameters

## 4.3 Turbo code error performance analysis

In this section, the results of the analysis of turbo codes given in terms of *BER* and the *Eb/No* were based on

- The effect of the number of iterations
- The effect of frame length
- The effect of the decoding algorithm chosen

### 4.3.1 The effect of the Number of iterations on Turbo Code performance

To show the effect of decoding iterations on turbo code performance, the program was simulated using a constant frame size of 4800 while varying the number of decoding iterations. The SNR range used was between 0 to 7 dB and the number of decoding iterations chosen for each transmission were [2, 4, 8 … 16].

**a. Simulation using log MAP algorithm:**

✓ ***LTE standard***

The Figure4.4 given below shows the BER for each iteration using the Log Map algorithm. According to the graph, the BER decreases with the increase in the number of iterations. For instance, for 16 iterations and for SNR of 2dB, the BER achieved is 0.014313 as shown in the

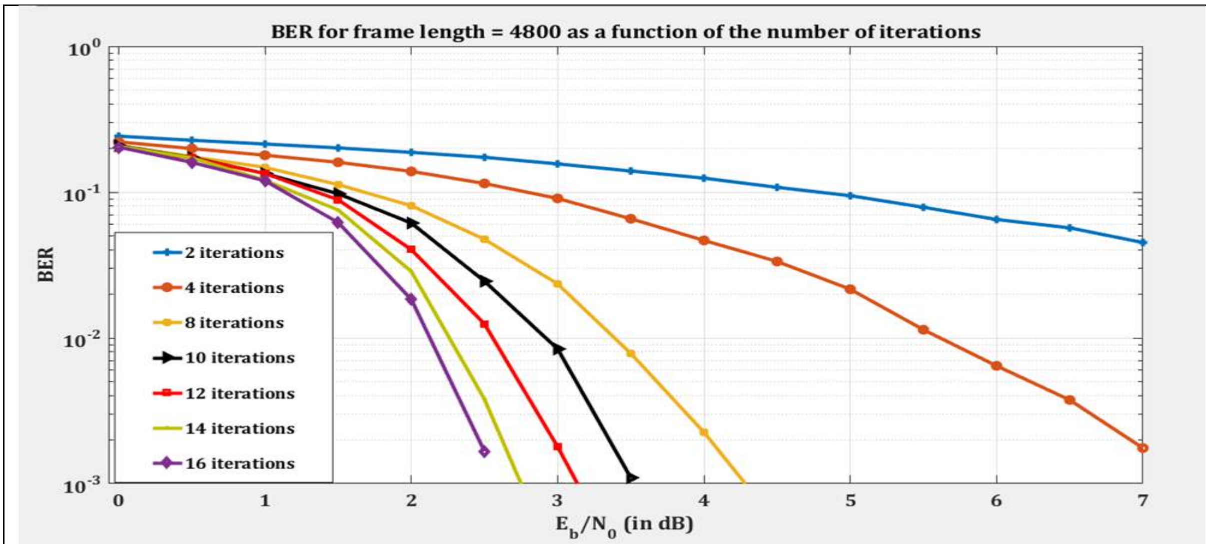



Figure 4.4 BER for frame length of 4800 while varying the number of iterations (LTE)

table 4.3. As the SNR increases, the BER decreases with less number of iterations. In fact, for an SNR greater than 4dB, only 10 iterations are needed to achieve a BER of $10^{-3}$.

| Iterations | BER at *Eb/No= 2dB* |
|---|---|
| 2 | 0.186021 |
| 4 | 0.135083 |
| 8 | 0.075958 |
| 10 | 0.061333 |
| 12 | 0.040313 |
| 14 | 0.028563 |
| 16 | 0.014313 |

Table4.3 BER at Eb/No=2dB for each iteration using Log MAP algorithm (LTE)

- ✓ ***WiMAX standard***

The Figure 4.5 given below shows the BER for each iteration using the Log Map algorithm. According to the graph, the BER decreases with the increase in the number of iterations. For instance, for 16 iterations and at the SNR of 2dB, the BER achieved is 0.017146 as shown in the table 4.4. As the SNR increases, the BER decreases with less number of iterations. . In fact for an SNR greater than 6dB, only four iterations are needed to achieve a BER less than $10^{-3}$.

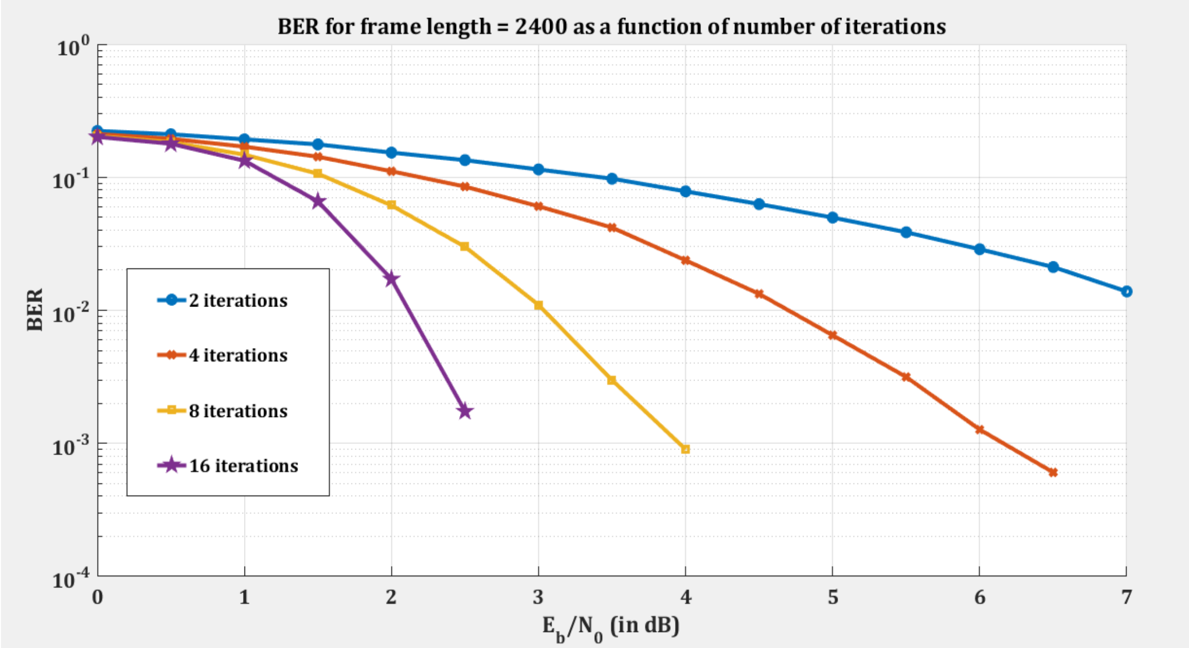



Figure 4.5 BER for different frame length while varying the number of iterations (WiMAX)

| Iterations | BER at *Eb/No= 2dB* |
|---|---|
| 2 | 0.153104 |
| 4 | 0.110792 |
| 8 | 0.061417 |
| 16 | 0.017146 |

*Table 4.4* BER at Eb/No=2dB for each iteration using Log MAP algorithm (WiMAX)

**b. Simulation using Max log MAP algorithm**:

✓ ***LTE standard***

The Figure4.6 given below shows the BER for each iteration using the Max Log Map algorithm.According to the graph, the BER decreases with the increase in the number of iterations. For instance, for 16 iterations and at the SNR of 2dB, the BER achieved is 0.018375 as shown in the table 4.5.

As the SNR increases, the BER decreases with less number of iterations. In fact, for an SNR of 7dB, only four iterations are needed to achieve a BER around $10^{-3}$.

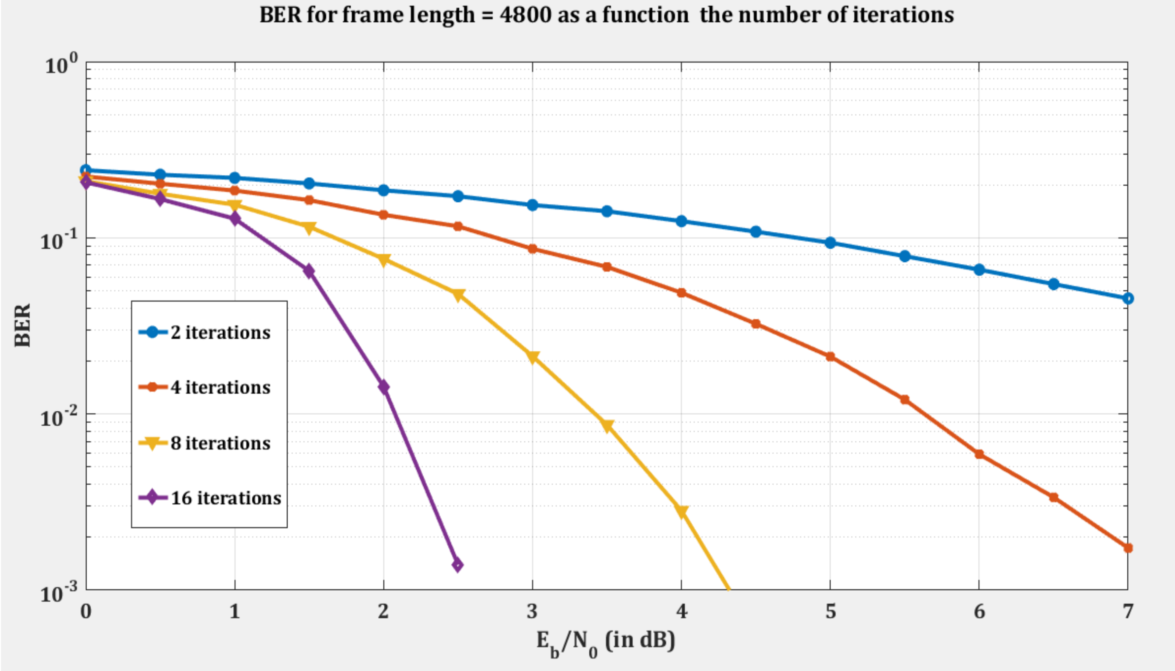



*Figure4.6: BER while varying the number of iterations using Max LOG MAP algorithm*

| Iterations | BER at *Eb/No= 2dB* |
|---|---|
| 2 | 0.187354 |
| 4 | 0.138771 |
| 8 | 0.080646 |
| 16 | 0.018375 |

Table 4.5 BER at Eb/No=2dB for each iteration using Max Log MAP algorithm(LTE)

*Remark*

Turbo codes are iterative codes, the exchange of the same information between the decoders result to better code performance.

### 4.2.2 The effect of frame length on Turbo Code performance using log MAP algorithm.

To show the effect of frame length on turbo code performance, the program was simulated using a constant number of 8 iterations at the decoder while varying the length of the frame.

- ✓ **LTE standard**

The Figure 4.7 given below shows the BER for each frame length used with eight decoding iterations. According to the graph, the BER decreases with the increase in size of the frame. For instance, at SNR=4.0dB, the BER of 2.02 x $10^{-3}$ is achieved for the largest frame length

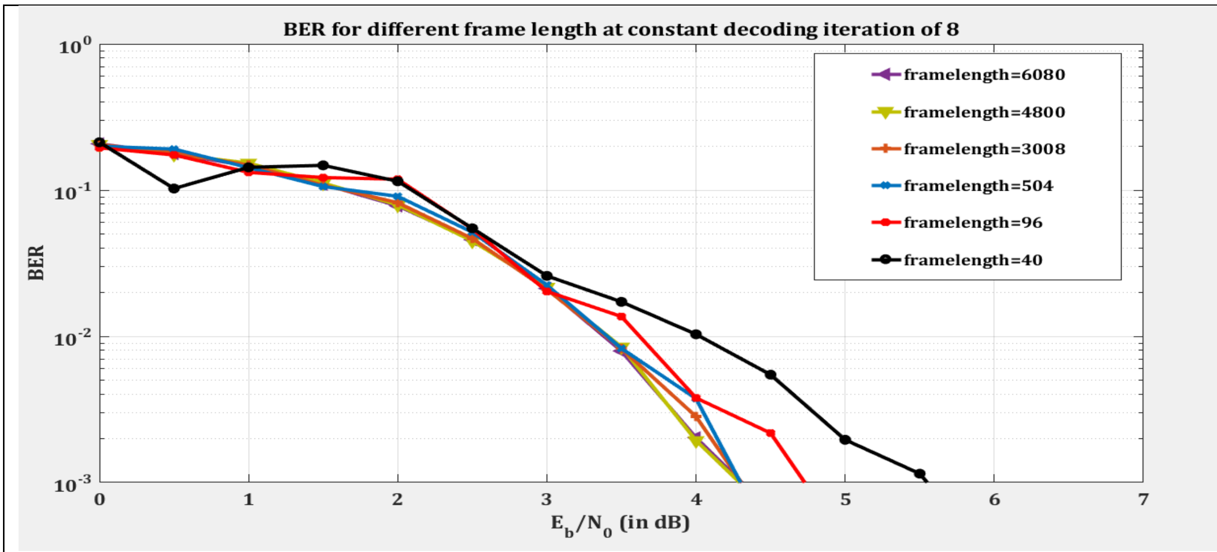



*Figure4.7:BERwhilevarying the frame length (LTE)*

of 6080 as shown in table 4.6

| Frame length | BER at EbNo=4dB |
|---|---|
| 6080 | 0.002023 |
| 4800 | 0.001938 |
| 3008 | 0.002826 |
| 504 | 0.003748 |
| 96 | 0.003788 |
| 40 | 0.010321 |

Table 4.6 BER at EbNo=4dB for each frame length (LTE)

✓ ***WiMAX standard***

The Figure4.8 given below shows the BER for each frame length used with eight decoding iterations.According to the graph, the BER decreases with the increase in size of the frame. For instance, at EbNo=4.0dB, the BER of 0.000530 is achieved for the largest frame length of 2400 as shown in table 4.7

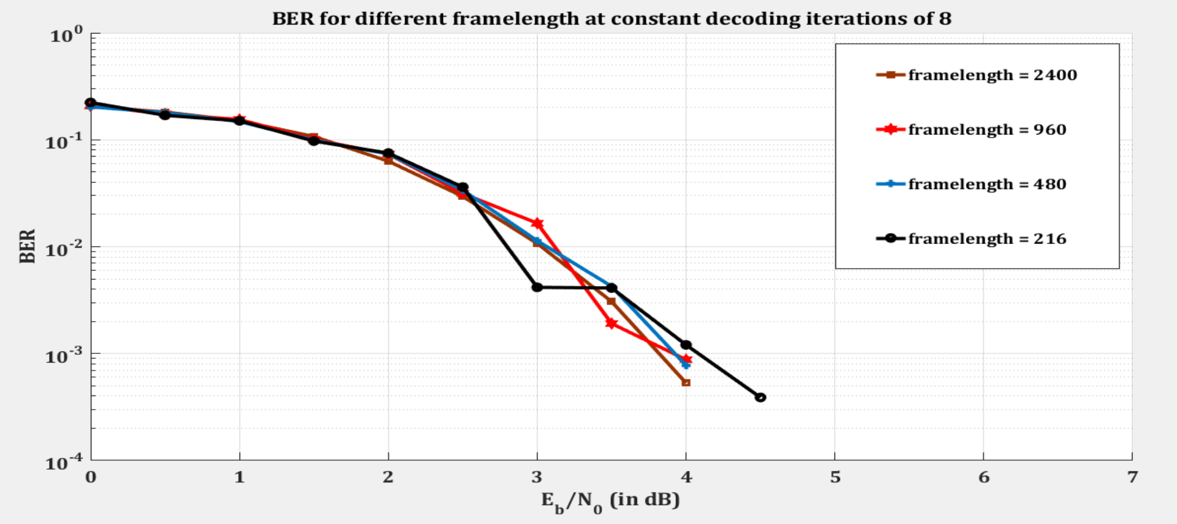



*Figure4.8: BER while varying the frame length (WiMAX)*

| Frame length | BER at EbNo 4dB |
|---|---|
| 2400 | 0.000530 |
| 960 | 0.000877 |
| 480 | 0.000774 |
| 216 | 0.001206 |

Table 4.7 BER at EbNo=4dB for each frame length (WiMAX)

*Remark:* Larger frame sizes means more latency as the encoding and decoding is done per frame. The performance improvement is achieved at the cost of increased latency.

### 4.2.3 The effect of decoding algorithm on Turbo Code performance

The two algorithms Log MAP and Max Log Map algorithms were compared in terms of number of iterations and the execution time.

a. **BER performance per iteration**

| Iterations | BER at *Eb/No= 2dB* <u>*Max Log MAP*</u> | BER at *Eb/No= 2dB* <u>*LOG MAP*</u> | Relative error % |
|---|---|---|---|
| 2 | 0.187354 | 0.186021 | 0.71148734 |
| 4 | 0.138771 | 0.135083 | 2.65761578 |
| 8 | 0.080646 | 0.075958 | 5.81305954 |
| 16 | 0.018375 | 0.014313 | 22.1061224 |

Table4.8 BER at Eb/No=2dB for each iteration for LTE standard

From table 4.8, one can see that the log MAP algorithm performs better as the number of iterations increases. This confirms the theoretical comparison between the two algorithms of section 3.2 of chapter 3

### b. Execution time in decoding per frame

The Figure 4.9 given below shows the Execution time against frame length for log MAP and

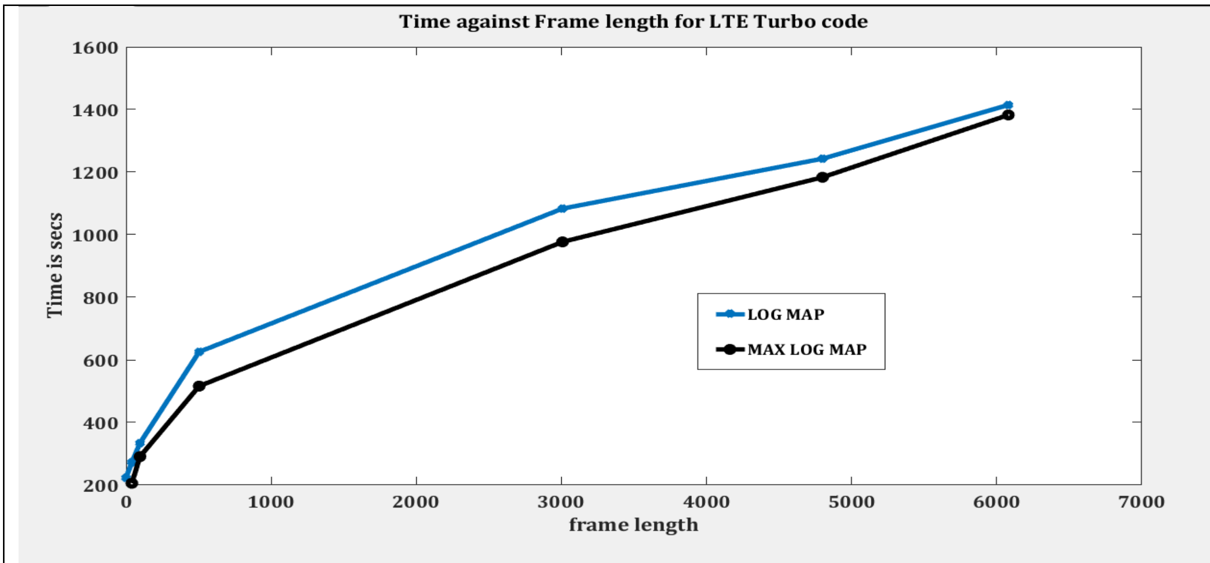



*Figure 4.9 Execution time against frame length*

MAX log MAP algorithm.

| Decoding algorithm | Frame length | Time |
|---|---|---|
| Log MAP | 4800 | 1242.81 |
| MAX Log MAP | 4800 | 1183.42 |

Table 4.9 time performance for the Log MAP and Max Log MAP (LTE)

According to figure 4.9, the decoding delay is large in Log-MAP algorithm compared to Max Log MAP algorithm. For instance,  the time taken to decode a frame size of 4800 using Log MAP algorithm was 1242.81s while using Max Log MAP algorithm was 1183.42s; a difference of 59.39s.

Log MAP algorithms offers the best BER performance with iterative decoding though it is slower as compared to Max log MAP algorithm. This is due to the negligence of the correction function in Max Log MAP algorithm that makes it fast, at a cost of a slight degradation of the BER

## 4.3 Simulation using an image

To see the performance of turbo codes as an iterative error correcting code, a Matlab picture ‘cameraman’ was used as a subject for the simulation .

Fig 4.10 image of cameraman

The image was first converted to binary image then sent through the channel of various SNR values , received and decoded using a couple of iterations.

To view the effect of coded and non-coded image, two simulations had been done.

- Simulation for uncoded binary image
- Simulation for coded binary image using turbo codes

### 4.3.1 Simulation of uncoded binary image

The binary image was sent through the channel without coding and decoded at the receiver using a hard decision technic. The following figures shows the results obtained of the received uncoded image at different SNR

**SNR=1dB**


Figure4.11a received uncoded image with SNR=1dB

a. **SNR=2dB**

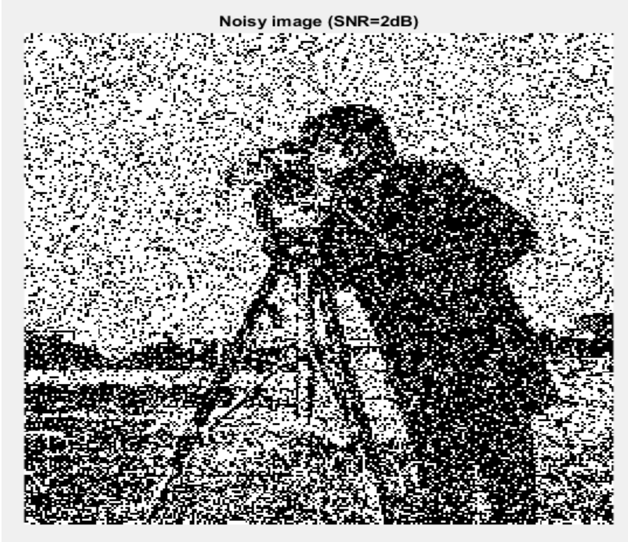



Figure4.11b received uncoded image with SNR=2dB

**b. SNR=3dB**

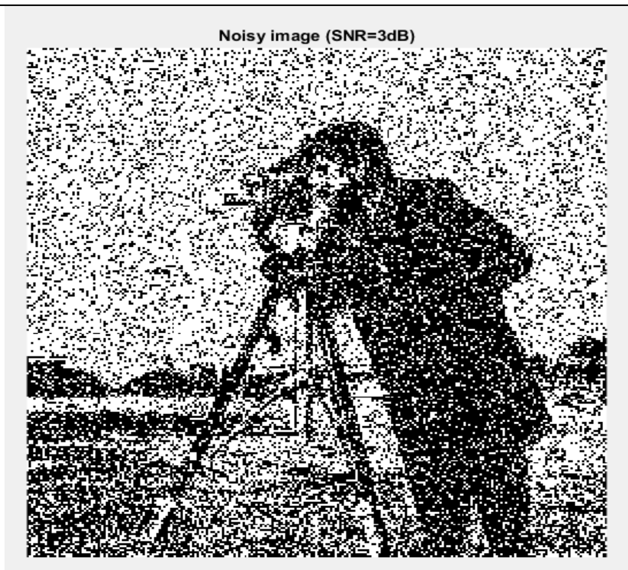



Figure4.11c received uncoded image with SNR=3dB

**Observation**

Though we were able to receive the picture sent through the channel, it is not as clear as the original picture we sent. The received image had a lot of noise.

Due to such cases, there was need for channel coding

### 4.3.2 Simulation of coded binary image

In this simulation, the binary image was first coded using turbo code and then sent through the channel and later on decoded at the receiver using Max log MAP algorithm. The following figures shows the iterative decoding of turbo codes at different SNRs with 1, 4, 8 and 16 iterations

**a. SNR = 1dB**


Figure 4.12a received iterated turbo coded image SNR=1dB

**b. SNR=2dB**

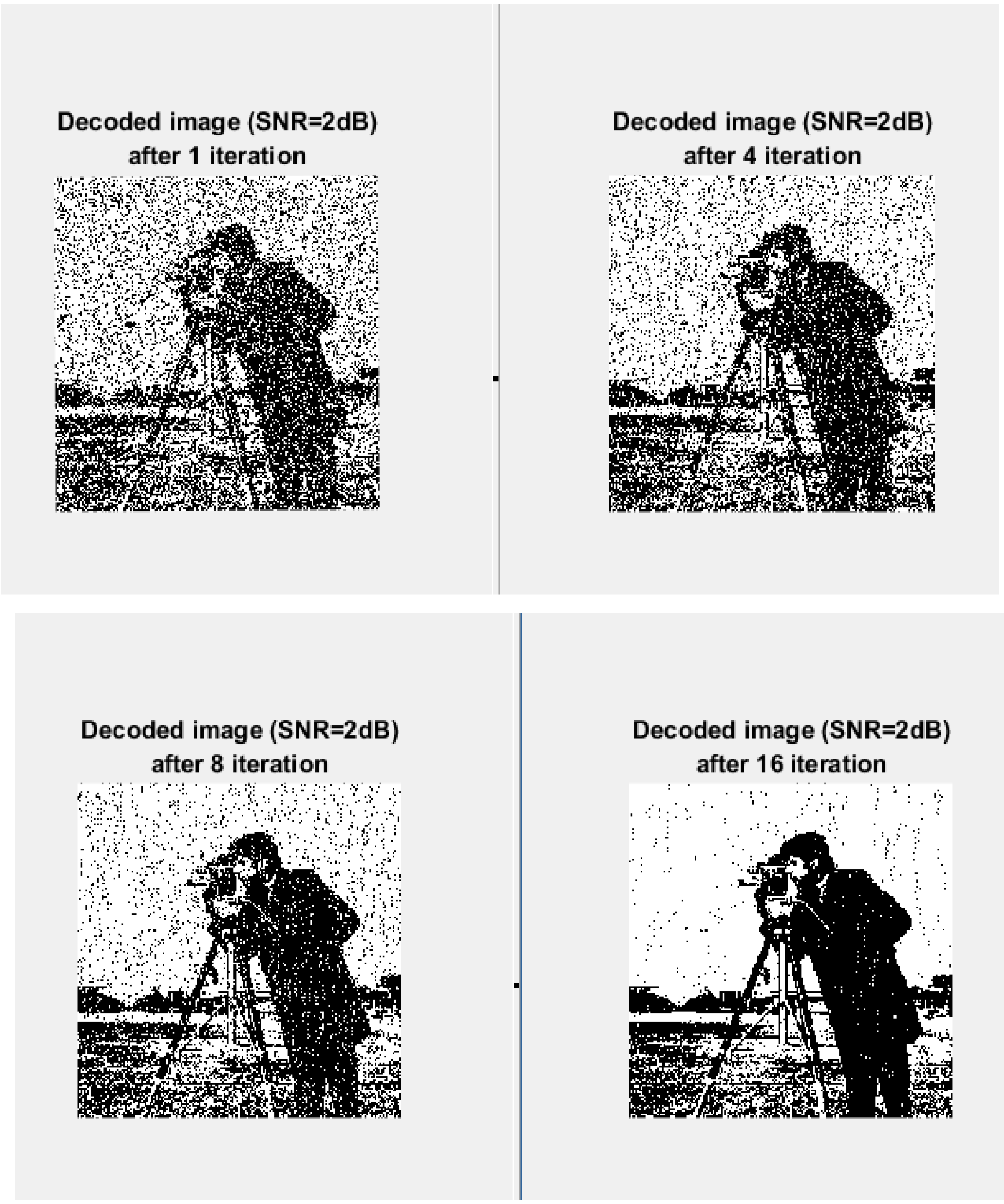



Figure 4.12b received iterated turbo coded image SNR=2dB

**c. SNR= 3dB**

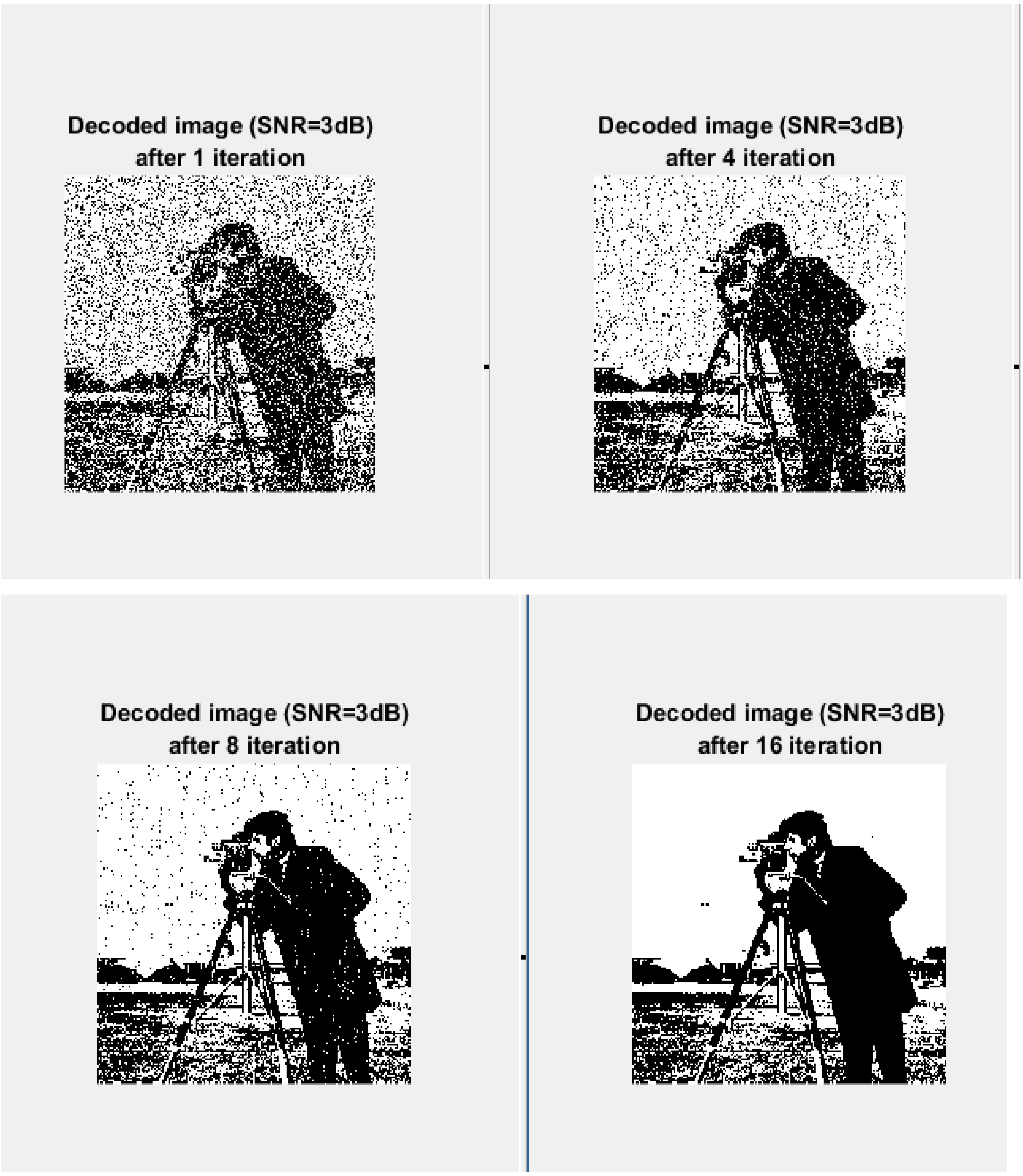



Figure 4.12c received iterated turbo coded image SNR=3dB

Observation

- The image got better after every iteration for any given value of SNR. The more the iterations are used, the clearer is the final image but the longer it takes to decode.

To quantify the effect of noise with the values of SNR and the number of iterations, a calculation of BER has been done and the results are shown in Table 4.10. It can be seen that the reduction of BER with the increase in number of iterations for different values of SNR is in accordance with the quality of the pictures. For instance, for SNR of 1dB, the BER is minimum with 16 iterations. By varying the SNR from 1dB to 3dB, one can also remark that the BER decreases as the SNR increases for any given iteration in the table.

| | Iteration =1 | Iteration =4 | Iteration =8 | Iteration =16 |
|---|---|---|---|---|
| SNR=1dB | 0.2278 | 0.1829 | 0.1498 | 0.1262 |
| SNR=2dB | 0.2054 | 0.1332 | 0.0780 | 0.0150 |
| SNR=3dB | 0.1847 | 0.0915 | 0.0227 | 0.0001 |

Table4.10 BER results for a given SNR and the iterations in image decoding

## 4.4 Conclusion

From the simulations, we can conclude that, the performance of the turbo code depends on a number of parameters. The performance of turbo code increases with

- The increase in the size of the frame
- The increase in the number of decoder iterations,
- The increase in the channel SNR

Thechoice of the decoder algorithm used in decoding also affects its performance in the sense that Log MAP algorithms offers the best BER performance with iterative decoding though it is slower as compared to Max log MAP algorithm

# GENERAL CONCLUSION

Turbo codes are used in Telecommunications applications such as:

- 3G and 4G mobile telephony standards; e.g., in HSPA and LTE.
- Terrestrial mobile television systems.
- The interaction channel of satellite communication systems, such as DVB-RCS
- IEEE 802.16 (WiMAX), a wireless metropolitan network standard.

This is because they show extraordinary performance at low SNRs close to the Shannon limit.

The simulation results showed that the performance of the turbo code depend on a number of parameters such as; the frame size, number of decoder iterations, the SNR and the choice of decoder algorithm.

The performance of the turbo code increases with the increase in the interleaver size (frame size), the number of iterations and the SNR. The choice of the decoder algorithm used in decoding also plays a major role in its performance in the sense that Log MAP algorithms offers the best BER performance with iterative decoding compared to Max Log MAP and SOVA algorithms but at the cost of the execution time.

The main disadvantage while using Turbo code as an error correcting code is the effect of increased latency during decoding to get better results. The introduction of Max log MAP algorithm has greatly helped to reduce the effect of time but it comes at a cost of a little BER degradation.

As for further improvements in turbo code performance, research should be focused in improving decoder performance and its complexity

# ANNEXES

ANNEXES

In this section, calculations of MAP algorithm are given [3]

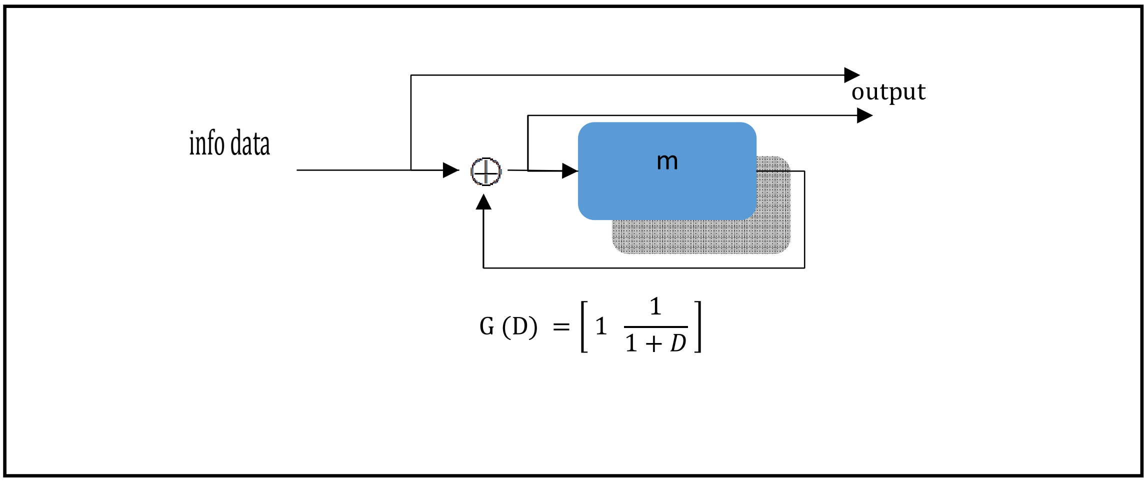
Assume $\frac{Es}{No}$ = ¼

The received BPSK modulated sequence r is given by

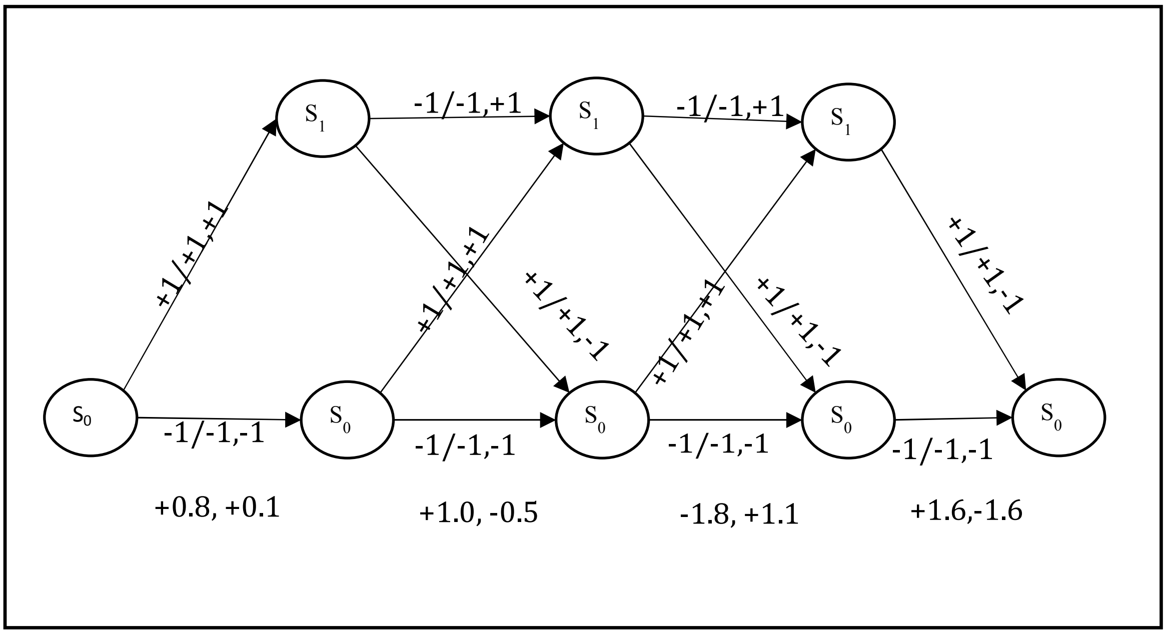
$r_i = x_i + n_i$

$r_i = (r_0, r_1, r_2, r_3, r_4 \ldots r_i)$

= (+0.8, +0.1, +1.0, -0.5, -1.8, +1.1, +1.6,-1.6)




A C(2.1.1) convolutional encoder




Trellisdiagram of a C(1.2.1) convolutional encoder

## ANNEXES

Calculations of the state metrics are given below

Branch state metric (Gamma-γ)

$$\gamma(s', s) = P(u_k) e^{u_k \frac{L(u_k)}{2}} e^{-\frac{Es}{No} 2 r_k v_k}$$

Assuming $P(u_k) = \frac{1}{2}$

$$\gamma(s', s) = K_i e^{-\frac{Es}{No} 2 r_k v_k}$$

$$\gamma(s_0, s_0) = e^{-\frac{Es}{No} 2 r_k v_k}$$

r=(+0.8, +0.1) and v=(-1,-1)

$$\frac{Es}{No} 2 r_k v_k = 0.8 \times (-1) + 0.1 \times (-1) = -0.45$$

$$\gamma(s_0, s_0) = e^{-\frac{Es}{No} 2 r_k v_k} = 0.6376$$

$$\gamma_0(s_0, s_1) = e^{-\frac{Es}{No} 2 r_k v_k}$$

r=(+0.8, +0.1) and v=(+1,+1)

$$\frac{Es}{No} 2 r_k v_k = 0.8 \times (+1) + 0.1 \times (+1) = 0.45$$

$$\gamma_0(s_0, s_1) = e^{-\frac{Es}{No} 2 r_k v_k} = 1.5683$$

$$\gamma(s_0, s_0) = e^{-\frac{Es}{No} 2 r_k v_k} = 0.6376$$

$$\gamma_0(s_0, s_1) = e^{-\frac{Es}{No} 2 r_k v_k} = 1.5683$$

$$\gamma_1(s_0, s_0) = e^{-\frac{Es}{No} 2 r_k v_k} = 0.7788$$

$$\gamma_1(s_0, s_1) = e^{-\frac{Es}{No} 2 r_k v_k} = 1.2840$$

$$\gamma_1(s_1, s_0) = e^{-\frac{Es}{No} 2 r_k v_k} = 0.4724$$

$$\gamma_1(s_1, s_1) = e^{-\frac{Es}{No} 2 r_k v_k} = 2.1170$$

$$\gamma_2(s_0, s_0) = e^{-\frac{Es}{No} 2 r_k v_k} = 1.4191$$

$$\gamma_2(s_0, s_1) = e^{-\frac{Es}{No} 2 r_k v_k} = 0.7047$$

$$\gamma_2(s_1, s_0) = e^{-\frac{Es}{No} 2 r_k v_k} = 4.2631$$

$$\gamma_2(s_1, s_1) = e^{-\frac{Es}{No} 2 r_k v_k} = 0.2346$$

Forward state metric (Alpha-α)

$$\alpha_k(s) = \sum_{s'} \gamma_{k-1}(s', s) \alpha_{k-1}(s')$$

$$\alpha(0) = \begin{cases} 1, & \text{all zero state of the encoder} \\ 0, & \text{not all zero state} \end{cases}$$

$$\alpha_1(s_0) = \alpha_0(s_0)\gamma_0(s_0, s_0) = 0.6376$$
$$\alpha_1(s_1) = \alpha_0(s_0)\gamma_0(s_0, s_1) = 1.5683$$

$$\alpha_2(s_0) = \alpha_1(s_0)\gamma_1(s_0, s_0) + \alpha_1(s_1)\gamma_1(s_0, s_1) = 3.8167$$
$$\alpha_2(s_1) = \alpha_1(s_0)\gamma_1(s_1, s_0) + \alpha_1(s_1)\gamma_1(s_1, s_1) = 1.5595$$

$$\alpha_3(s_0) = \alpha_2(s_0)\gamma_1(s_0, s_0) + \alpha_2(s_1)\gamma_1(s_0, s_1) = 5.7821$$
$$\alpha_3(s_1) = \alpha_2(s_0)\gamma_1(s_1, s_0) + \alpha_2(s_1)\gamma_1(s_1, s_1) = 9.3379$$

Backward state metric (Beta-β)

$$\beta_{k-1}(s') = \sum_{s} \gamma_{k-1}(s', s)\beta_k(s)$$

$$\beta(0) = \begin{cases} 1, & terminated\ trellis \\ 0, & not\ terminated\ trellis \end{cases}$$

$$\beta_3(s_0) = \beta_4(s_0)\gamma_3(s_0, s_0) = 1.0$$
$$\beta_3(s_1) = \beta_4(s_0)\gamma_3(s_1, s_0) = 4.9530$$

$$\beta_2(s_0) = \beta_3(s_0)\gamma_2(s_0, s_0) + \beta_3(s_1)\gamma_2(s_0, s_1) = 21.3497$$
$$\beta_2(s_1) = \beta_3(s_0)\gamma_2(s_1, s_0) + \beta_3(s_1)\gamma_2(s_0, s_1) = 4.9095$$

$$\beta_1(s_0) = \beta_2(s_0)\gamma_1(s_0, s_0) + \beta_2(s_1)\gamma_1(s_0, s_1) = 20.4790$$
$$\beta_1(s_1) = \beta_2(s_0)\gamma_1(s_1, s_0) + \beta_2(s_1)\gamma_1(s_1, s_1) = 31.2365$$

$$\mathrm{L}(u_r) = \ln \frac{\sum_1^+ \alpha_{k-1}(s')\gamma_.(s', s)\beta_k(s)}{\sum_1^- \alpha_{k-1}(s')\gamma_.(s', s)\beta_k(s)}$$

$$\mathrm{L}(u_0) = \ln \frac{\alpha_0(s_0)\gamma_0(s_0, s_1)\beta_1(s_0)}{\alpha_0(s_0)\gamma_0(s_0, s_0)\beta_1(s_0)} = 0.4778$$

$$\mathrm{L}(u_1) = \ln \frac{\alpha_1(s_0)\gamma_1(s_0, s_1)\beta_2(s_0) + \alpha_1(s_1)\gamma_1(s_1, s_0)\beta_2(s_0)}{\alpha_1(s_0)\gamma_1(s_0, s_0)\beta_2(s_0) + \alpha_1(s_1)\gamma_1(s_1, s_1)\beta_2(s_1)} = 0.6154$$

$$\mathrm{L}(u_2) = \ln \frac{\alpha_2(s_0)\gamma_2(s_0, s_1)\beta_3(s_0) + \alpha_2(s_1)\gamma_2(s_1, s_0)\beta_3(s_0)}{\alpha_2(s_0)\gamma_2(s_0, s_0)\beta_3(s_0) + \alpha_2(s_1)\gamma_2(s_1, s_1)\beta_3(s_1)} = -1.0301$$

Hard decision

$$\mathrm{L}(u_r) = \begin{cases} 1, & x > 0 \\ 0, & x < 0 \end{cases}$$

$$\hat{u} = (1, 1, 0)$$

1. Jakob L. Buthler, Troels Jessen, Michael Buhl, Rune Simonse (Group 11gr850), "*Turbo Codes and OFDM Implementation for LTE mobile systems*" Aalborg University, USA; May 2011

2. Branka Vuceic and Jinhong Yuan;"*Turbo Codes Principes and applications";* Kluwer Academic Publishers,USA; 2000.

3. Prof.Adrish Banerjee "*An introduction to coding theory";*Kapur university of IIT, India. https://www.youtube.com/channel/UCtdbWilktCfgHGks-HHm9QQ/videos

4. H. Meliani, A.Guellal "*Comparison Between Viterbi Algorithm Soft And Hard Decision Decoding"; * University of Blida, Algeria

5. CC Lab, EE, NCH "*Coding and Communication Laboratory,* CHINA", National Chung Hsing University, China.

6. Claude Berrou; *"Codes and Turbo Codes*" (translated from French by Janet Ormrod); Springer,Paris; 2007

7. Daniel J Costello, Jr, Oscar Y Takeshita, Hermano A. Cabral, "*Some thoughts on the equivalence of systematic and Non-systematic convolutional Encoders"* university of Notre Dame, Ohio state university August,2,2001

8. Anum Imran, "Software Implementation and Performance of UMTS Turbo CODES," Tampere University of Technology, Masters of Science Thesis, pp.15, March 2013.

9. Silvio A. Abrantes;*"From BCJR to turbo decoding: MAP algorithms made easier";* University of Kansas,USA; April 2004.

10. Jagdish D. Kene, K.D. Kulat*"Soft Output Decoding Algorithm for Turbo Codes Implementation in Mobile Wi-Max Environment.*"2nd International Conference on Communication, Computing & Security [ICCCS-2012]. https://www.researchgate.net/publication/257744000