Simulation of Serial Concatenated Convolutional Coded Turbo Decoder System

Lincoln J. Sadler[1]

University of New South Wales at the Australian Defence Force Academy

The modern world has a large reliance upon the quick, accurate transfer of information; it is involved in almost every facet of society. For this transfer to be achieved it requires the information to be coded. This paper will look at the development of a simulation of a SCCC turbo code system, one of the first of these codes to approach the theoretical data transfer limit. It will provide a background on the underlying theories and a breakdown of the development procedure of the simulation. A look at incremental systems developed in the process will show the potential coding gains that come from error control coding.

Contents

I. Introduction 2
II. Background 2
III. Project description 5
IV. Development Process 5
A. Manual Analysis 5
B. Initial Programming 5
C. Turbo Code: Encoder and Decoder 6
V. Conclusions 6
VI. Recommendations 7

Acknowledgements 7
References 7

Nomenclature

BER	=	Bit Error Rate
BPSK	=	Binary Phase Shift Keying
E_b/N_o	=	Energy per information bit to noise power spectral density ratios
FEC	=	Forward Error Correction
HCCC	=	Hybrid Concatenated Convolutional Codes
MAP	=	Maximum A Posteriori
PCCC	=	Parallel Concatenated Convolutional Codes
SCCC	=	Serial Concatenated Convolutional Codes
SNR	=	Signal to noise Ratio

[1]Z3370309 PLTOFF, School of Engineering& Information Technology.ZEIT4501

I. Introduction

In the modern world communicating via the transmission of data is the life blood of society, with modern mobile devices and smart homes it has become integrated into the most basic of tasks. The revolution that allowed this to happen was the rise of digital communication systems. Coding techniques that allow for error free or low error transmissions at high speed are fundamental to these communication systems. In the 1940's Shannon's work showed that for any given channel there is a maximum rate at which information can be reliably transmitted across it. It was long thought that this would require an infinitely long message transmitted in infinite permutations to achieve but the introduction of turbo codes in 1993 showed that the Shannon limit was attainable by coming within 0.5dB of it. The original turbo codes used a parallel configuration but due to the performance of this code it was rapidly expanded to other code types and iteration schemes [1].

The variety of new coding schemes based on turbo codes has allowed communications technology to achieve things that were not thought possible before. The codes are used in many applications such as communicating with deep space satellites, in mobile phones and in hard drives to ensure efficient error free data transfer.

An investigation of the background theory of the codes and an implementation of a serial concatenated turbo code for simulation to show the coding gain of this technique will be carried out in this paper.

II. Background

The most basic communication system consists of a transmitter, a communication medium (channel) and a receiver. The quality of the channel can cause a number of effects such as attenuation, distortion, interference and noise, these effects can reduce the probability that the transmitted message will be received error free[2]. The effects can be divided up into two main types' additive noise and channel perturbations. Additive noise will generally come from components in the receiver and can be called thermal noise. To reduce these effects upon the message an error control coding scheme is used.As the system we are looking at is a one way transmission we are using a form of Forward Error Correction (FEC). A forward error correcting code has enough redundancy added to the data that it is possible to correct errors in the data without referring back to the transmitter [2].

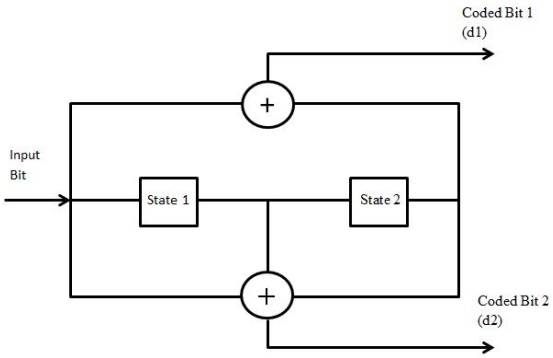

Figure 1: Rate 1/2, (5, 7) Convolutional Encoder

Convolutional encoders add the redundancy to the message to provide the FEC. They consist of a number of different shift registers or memory slots that are tapped in various places to provide the required code. In this project we used a non-systematic G[5, 7]convolutional encoder, a block diagram of this coding scheme is in Figure 1. This is a rate ½ encoder, meaning that for each bit input two coded bits are output doubling the size of the message. The encoder performs a modulo 2 or exclusive-OR operation on the values at the points where it is tapped to determine the output values. In the example there are three possible tap points, the input bit, the output of state 1 and the output of state 2. The position of the taps are what determines the descriptive numbers in the brackets, a 101 for coded bit 1 which equals 5 in base 8 and 111 for coded bit 2 which equals 7 in base 8 [3]

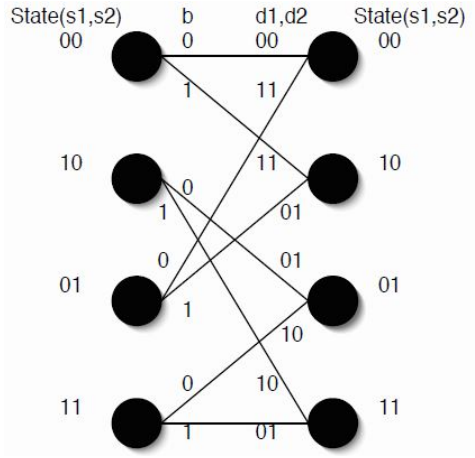

Figure 2: Trellis Diagram of Convolutional Encoder [4]

Convolutional encoders are also called trellis encoders as their encoded message can be represented as a trellis diagram which shows the progression from state to state for each input bit and the corresponding output coded bit pair. A trellis diagram as in Figure

2 has each possible state on the vertical axis with each state connected to the next state by allowable transitions for that state. Due to the structure of the encoder there is a constraint upon which state an originating state can transition to next, for example if the encoder was in state (00) it can only transition to state (00) for a 0 input bit or state (10) for a 1 input bit, it is not possible for it to transition from state (00) to state (01) or state (11). It is this constraint that allows the decoding of the encoded message. When discrete time is used as a horizontal access in a trellis diagram it is possible to track the possible paths through the trellis for each transition as new bits arrive.

The problem with just making a hard decision on each bit and tracking back through the trellis (as done with a Viterbi Decoder) is that decisions are made early and information is lost in the process. When noise is added to a signal it can cause a binary 1or0 tobe mistaken for the other when a hard decision is made based upon a set decision threshold level. In Figure 3 two signals used to represent a 1 and a 0 bit are shown. In the first case there is no noise, in the second and third cases there is differing signal strength to noise strength (SNR) values. It can be seen from the different normal distribution curves that as the SNR goes up the variance goes down and errors based on hard decision threshold are less likely [5]. For lower SNR the signal is more spread and there is a greater chance of errors, to reduce this chance soft-decision decoding can be used

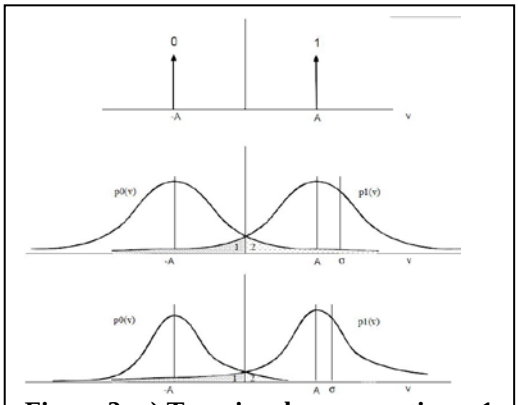

Figure 3: a) Two signals representing a 1 and 0 bit. b) SNR =2. c) SNR=4[5]

The Maximum A Posteriori (MAP) decoding algorithm is the optimal method for decoding a convolutional code. It can be used to determine the path of the message through the trellis, as it is a maximum likelihood decoding algorithm, it determines the probability that a 0 was decoded given a 1 was transmitted; this probability is a function of the area under the crossed parts of the normal curves. Once the probability of the decoded bit has been determined a hard-decision can then be finally made as to the actual received bits value.

The MAP Decoding algorithm is also called the forward-backward algorithm as it transverses the trellis in both directions during its calculations. The decoder first calculates the probabilities for each coded bit based upon the variance of the noise on the channel, for this project it was assumed that the noise would only be Additive White Gaussian Noise (AWGN) this allowed the calculation for the coded bit metrics to be as in (1). In this calculation the value of $y_{j,n`}$ is the received value, $d_{j,n`}$ is either +1 or -1, σ^2 is the variance of the noise. For each received bit a metric is worked out for $d_{j,n`}$ equal to both 1 or -1, the metric then gives us a probability of the received bit being a 1 or -1.

$$Metric\ for\ coded\ Bits = exp\left(-\frac{(y_{j,n`}-d_{j,n`})^2}{2\sigma^2}\right) \qquad (1)$$

The coded bit metrics are then used to calculate the probability for each transition for example in figure 2 the transition from state (00) to state (00) would use the coded bit metric for d1 = 0 and d2 = 0. The transition probability (γ) is the normalised sum of each possible transition. Once the γ's have been calculated the decoder moves forward through the trellis calculating the forward state probability (α), this calculation starts from state (00)as the encoder is reset between message blocks. This probability is determined by the normalised sum of the previous α_{j-1}probabilitymultiplied by the γ transition required to move from the α_{j-1} to the new α_j. In Figure 4 it can be seen that raw value for α_j (m) will be equal to (2), once all the α values for a transition have been calculated the are normalised.

$$\alpha_j(m) = \alpha_{j-1}(m_1') * \gamma_j(m_1', m) + \alpha_{j-1}(m_2') * \gamma_j(m_2', m) \qquad (2)$$

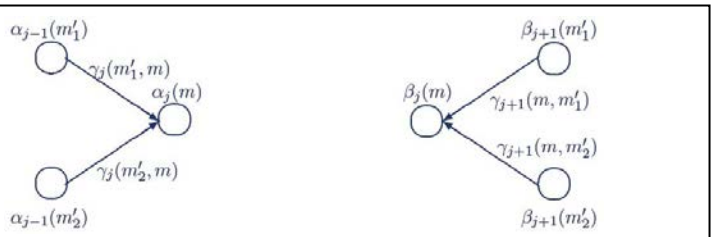

Figure 4: Representation of Alpha and Beta Calculations [4]

Once the trellis has been traversed fully forwards it is then traversed backwards to determine the reverse state probability (β). Depending upon the type of termination used on the message block the β values will either be initialized to 1 for state (00) and 0 for the rest or for all states equal to 0.25. If there is a terminating tail added to the message block the trellis will always end upon the state (00), if there is not then there is an equal chance to terminate on each state. The β_j value for the current transition is calculated by

normalising the sum of previous β_{j+1}valuesmultiplied by the γ transition required to move from the $\beta_{j+1}$to β_j. From Figure 4 it can be seen that the calculation for β_j (m) before normalisation will be as in (3).

$$\beta_j(m) = \beta_{j+1}(m_1') * \gamma_{j+1}(m, m_1') + \beta_{j+1}(m_2') * \gamma_{j+1}(m, m_2') \quad (3)$$

After the α, β and γ's have been calculated for the received message block they can be used to determine both the coded bit probabilities and the information bit probabilities for the message. These probabilities are the likelihood of the value of the coded bit or information bit based on the received values, the signal to noise ratio of the channel and the type of encoder used.

First proposed in 1993 turbo codes are a coding scheme that uses two convolutional encoders for encoding and two MAP decoders in the decoder to achieve a performance that comes within 1 dB of Shannon's limit. There are three different architectures that are used in their implementation parallel concatenated (PCCC), serial concatenated (SCCC) and hybrid concatenated (HCCC). Due to ease of implementation this project focused on the SCCC form of turbo codes, a block diagram of a SCCC system is in figure 5.A SCCC turbo code uses two convolutional encoders with a pseudo-random interleaver between them to encode the message, the interleaver has the effect of spreading out the message, and by not having sequential bits transmitted next to each other the system gains a measure of burst error protection. As there are two encoders the final size of the transmitted message block is four times the size of the original message.

The SCCC turbo decoder consists of a pair of MAP decoders that work cooperatively through the use of the a-posteriori output of the one decoder being used as a-priori input for the other decoder to increase the accuracy of their output. The inner decoder starts without initialisation information, but in subsequent iterations the soft-decision information from the outer decoder is used, after a number of iterations the output of the decoders will converge on a steady state output[6]. It is this feeding of information from one decoder to another that gives the turbo decoder its name as it resembles the feedback of the exhaust in an automobile turbo engine to increase performance. In Figure 5 it can be seen that each decoder has the ability to provide improved information regarding both the input and output of the corresponding encoder. The inner decoder corresponds to encoder 2 and can therefore provide a better estimation of the likelihood of data V or C_V, as the information regarding C_V is of no use to us that output is not used. The output of the inner decoder is de-interleaved to provide information about C_U that is used to initialise the outer decoder. The outer decoder corresponds to encoder 1 and both pieces of information provided by its output improve our estimation of the input message so they are both used. The probability of coded message C_U is interleaved and given to the inner decoder as initialisation probabilities for the message V which is the input of encoder 2. After enough of iterations around the loop for the soft-decision values to converge to stability the second output of the outer decoder, probabilities of the input message, are output to have a hard-decision made on them to give the decoded output.

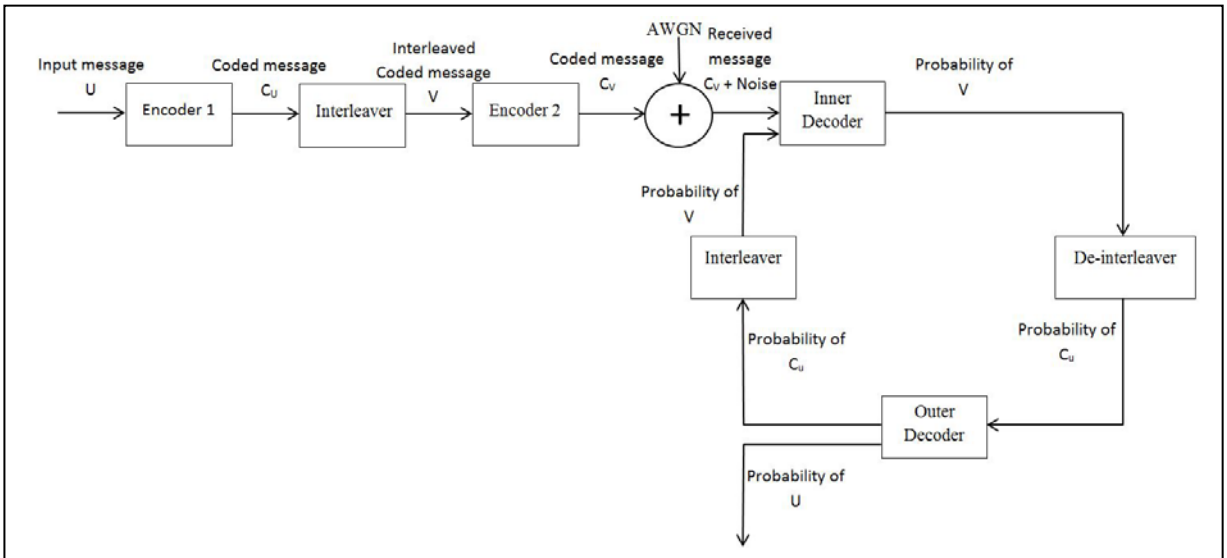

Figure 5: Block Diagram of SCCC system

The tool we used to assess the performance of a transmission system is the plot of the BER curve. This metric tells us the number of errors per bits for a given E_b/N_o. The E_b/N_o value describes the average amount of energy per information bit in comparison to the power density of the noise, as this value is expressed in dB a value of 3dB would indicate that the energy in the bit is twice that of the energy in the noise [7]. The BER tells us the amount of errors per bits in the signal a value of 10^{-3} as seen in Figure 6 means that for every 1000 bits there will be 1 error. In Figure 6 wecan see that for a BER of 10^{-3} the MAP decoder has an E_b/N_o value of 4.2dB while the BPSK has a E_b/N_o value of 6.8dB. This tells us that to achieve one error in one thousand bits the MAP

decoder requires the signal to have 2.5 times the energy of the noise while the BPSK decoder to achieve the same BER will require the signal to have 4.8 times the energy of the noise. The convolutional encoding and MAP decoding allows us to transmit the signal at approximately half the power of theBPSK which for a mobile device would mean a increased battery life, or we can transmit at the same power level but have increased range.

III. Project Description

The aim of this project was to investigate the theoretical and experimental characterisation of SCCC turbo decoders and their applications. It involved the development of a SCCC turbo decoder system simulation in MATLAB to demonstrate the potential coding gain from a turbo coding scheme. As a part of the development it was necessary to also develop convolutional encoders, pseudo-random interleavers and the associated de-interleavers, and MAP decoders.

IV. DevelopmentProcess

The development of this project consisted of a number of steps.The first was to manually work througha small MAP decoder example to ensure that a full understanding of the convolution and decoding process was gained. This second step used this understanding to convert the process into an algorithm for programming and a program was implemented. The third step was to modify a single convolutional encoder and MAP decoder to create both a SCCC turbo encoder and a SCCC turbo decoder.

A. Manual Analysis

To develop an understanding of the fundamentals of the MAP decoder a fully worked example for six channel symbols was developed. This involved working through the algorithm and showing the probabilities at each step. After we were fully satisfied with the understanding from the manual work through the encoder and MAP decoder from the example were then constructed in MATLAB and testing begun to ensure a correct output was achieved. When the encoder and decoder in MATLAB were shown to be operating correctly then the turbo decoder simulation was constructed using this algorithm as the core. Testing was then carried out to ensure proper integration of the blocks into the turbo decoder design.

The example used was for a rate 1/2, non-systematic G[5, 7]convolutional encoder as seen in figure 1. The data to be transmitted was a simple 3 bit message of [1, 0, 0]. The 6 bit output of the encoder then had AWGN added to it which simulates the noise from the receiver. The values for the received bits were then used to calculate the metrics for each bit transmitted, doing the calculations manually allowed the development of algorithms that could be transferred to a program. After the metrics were calculated they were used to calculate the α, β and γ values.

B. Initial Programming

Analysis of the procedure used to calculate the various probabilities in the MAP decoder allowed the development of algorithms for a programing implementation of the calculations.

Once the convolutional encoder and MAP decoder had been programmed in MATLAB they were tested individually to ensure they were operating correctly. The testing involved using the known values from the example to check that the outputs were correct. Once the programmed encoder and decoder were working correctly for the example increasing sized messages were used to check operation, due to the difficulty of manually checking the calculations for large bit sized messages values were chosen randomly and checked to ensure they were as expected based upon the surrounding values. To measure the performance of the decoder it was simulated at various energy per information bit to noise power spectral density ratios(E_b/N_o) to allow the calculation of a bit error rate (BER). The plot of this

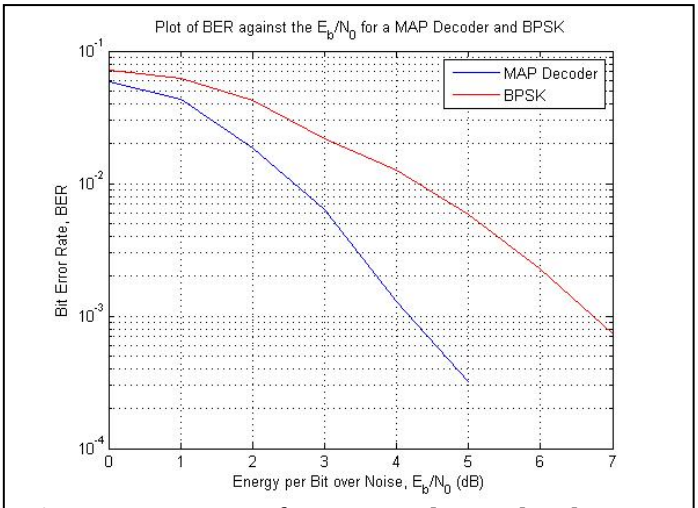

Figure 6: BER curves for BPSK and MAP decoder

BER was compared to the BER for a signal that was only subject to BPSK modulation is in Figure 6.It can be seen that just using a single convolutional encoder and MAP decoder paring can give significant performance gains, for example 2.5dB at BER = 10^{-3}.

C. Turbo Code: Encoder and Decoder

After confirmation that the encoder and decoder functions were working correctly development began on their incorporation into a turbo decoder system.The SCCC turbo system required a pseudo-random interleaver and de-interleaver for correct operation; this randomises the data between the encoding stages. To create them the randi function was used with a seed generated from the computer clock time, the seed was generated at the start of the function and passed to each subsequent interleaver and de-interleaver. The interleaver/de-interleaver pair was checked with both small and large message size to ensure that it was operating as planned.Once the interleaver was testedthe encoding section of thesimulation was programmed and assessed for correct operation, after each block the output of the previous block was checked for correctness and then input to the next block. The encoding and noise addition section of the simulation were now correct so the development of the decoder could proceed.

The development of the turbo decoder section required the modification of the MAP decoder block as the initial MAP decoder block did not have any a-priori input information. In the initial iteration of the turbo decoder there will be no a-priori information available from the outer decoder so an initial probability vector is created with the value set to a half (Probability = 0.5) where the binary values are equally likely and will have the same effect of there not being a-priori information while providing the variable name for the next iteration when it is required.The information being supplied to the inner decoder is a-priori information regarding the input bits to the second encoder; this information is used in the calculation of the γ values. In correct operation the inner decoder output will be biased by the a-priori information from the outer decoder leading to a more accurate estimation of the input bits. To test that a change in the a-priori probability provided an improvement in the MAP decoder performance a BER curve was generated from the inner decoder when it was supplied with a constant biasing probability. The biasing probability was then changed and another BER generated, four BER curves were generated and they are shown in Figure 7. It can be seen from Figure 7 that as expected with a higher probability the performance of the decoder was much better than with a lower probability. These curves show that the inner decoder is operating as desired the development then moved to the outer decoder.

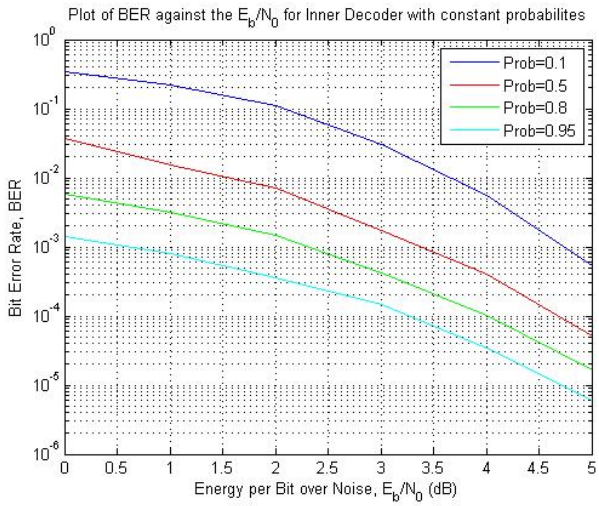

Figure 7: Plot Inner Decoder of BER curves for various biasing input

The a-priori information that is passed to the outer decoder is information regarding the coded output bits of the first encoder this information. The outer decoder does not use the received data values in its calculations of the γ values it instead uses the a-priori information passed to it by the inner decoder. Initial testing has been carried out on the outer decoder section of code but it has not yet been successful and due to time restraints it has not been made functional.

V. Conclusions

During the conduction of the development of this simulation various challenges were met, and while most were overcome, some were not. The goal of a fully functioning SCCC turbo decoder simulation was not achieved but incremental development along the way allowed data to be gathered to illustrate the potential for coding gains from a turbo system.

The development of a working simulation of a non-systematic, convolutional encoder and MAP decoder system allowed the determination of potential coding gains from this form of FEC.

The testing of the inner decoder showed the influence that small differences in a-priori information can have on the output of the decoder, for example at BER = 10^{-4} there is a difference of 1.5dB between a probability of 0.95 and 0.8.

Despite not achieving a functional SCCC turbo simulation an understanding of the core processes and procedures for the development of a system was gained.

VI. Recommendations

It is recommended that subsequent investigations be conducted within the following topics related to this project to further the understanding of the coding gains provided by a SCCC turbo system.

1. *Investigate the failure of the outer decoder to determine if it is a coding or algorithmic error;*
2. *The investigation and development of a puncturing code to improve the encoding rate.*

Acknowledgements

I would like to acknowledge and thank my supervisor, Mark Reed for the support and guidance he has given me, as well as the patience he has shown throughout the course of this project. I would like to also thank PhD student Amin Movahed for his timely technical advice and guidance through the challenges of developing a SCCC turbo system.

To my wife Bec for her patience, tolerance, love and supportwhileallowing me to work on this project and she dealt with the kids.

References

1. Soleymani, M., Gao, Y., & Vilaipornsawai, U. (2002). *Turbo Coding for Satellite and Wireless Communications.* Kluwer Academic Publishers.
2. Abrantes, S. (2004, April). *From BCJR to turbo decoding: MAP algorithms made easy.* Retrieved 2014, from Faculty of Engineering , University of Porto: http://paginas.fe.up.pt/~sam/textos/From%20BCJR%20to%20turbo.pdf
3. Reed, M. (2006, February). Statistical Communication Theory. Canberra.
4. Reed, M. (2005). Turbo Receiver Design: From Theory to Practice. *AusCTW 2005.* Brisbane.
5. Langton, C. (2014). *Tutorial 12 – Convolutional Coding and Decoding Made Easy.* Retrieved 2014, from http://complextoreal.com/: http://complextoreal.com/wp-content/uploads/2013/01/convo.pdf
6. Wade, G. (2000). *Coding Techniques: An introduction to compression and error control.* Palgrave.
7. Wolff, R. (2011). *Noise, S/N and E_b/N_o,* Retrieved 2014, from http://www.coe.montana.edu/: http://www.coe.montana.edu/ee/andyo/EE447/EB%20and%20NO.pdf
8. Benedetto, S., Divsalar, D., Montorsi, G., & Pollara, F. (1996, November). *A Soft-Input Soft-Output Maximum A Posteriori.* Retrieved 2014, from www.mit.edu: http://web.mit.edu/~6.962/www/www_fall_2000/jnl/divsalar-tm.pdf
9. Langton, C. (2014). *Tutorial 24b – The MAP decoding algorithm, step-by-step Part 2.* Retrieved 2014, from http://complextoreal.com/: http://complextoreal.com/wp-content/uploads/2013/01/turbo2.pdf
10. Langton, C. (2014). *tutorial-24a-turbo-coding-and-map-decoding-part-1*. Retrieved 2014, from http://complextoreal.com/: http://complextoreal.com/wp-content/uploads/2013/01/turbo1.pdf
11. Ryan, M., Frater, M., & Pickering, M. (2011). *Fundamentals of Communications and Information Systems.* Canberra: Argos Press.
12. Ryan, W. (2014, October). *Concatenated Convolutional; Codes and Iterative Decoding.* Retrieved 2014, from Technion, Electrical engineering: http://webee.technion.ac.il/people/sason/ryan_chapter.pdf
13. Skalar, B. (1997). *A primer on Turbo Code Concepts.* Retrieved 2014, from http://ieeexplore.ieee.org/Xplore/home.jsp: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=642838
14. Sweeney, P. (2002). *Error Control Coding: From theory to practice.* Chichester: John Wiley & Sons, Ltd.
15. Vashe, A. (2005). *Turbo Code Tutorial.* Retrieved 2005, from http://www.vashe.org/ http://www.vashe.org/turbo/turbo_primer_0.0.pdf