Spatially Coupled Codes in Turbo Equalization

MGENI MAKAMBI MASHAURI
MASTER´S THESIS
DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY
FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY

Spatially Coupled Codes in Turbo Equalization

Mgeni Makambi Mashauri
mg7107ma-s@student.lu.se

Department of Electrical and Information Technology
Lund University

Supervisor: Michael Lentmaier, Muhammad Umar Farooq

Examiner: Fredrik Rusek

17th June 2019

Printed in Sweden
Tryckeriet i E-huset, Lund

Abstract

In this thesis we investigate the application of spatial coupling in turbo equalization. We show that with spatial coupling we can have a fresh perspective on the trade-off between performance in the waterfall and error floor region often encountered in the choice of codes for turbo equalization.

Describing turbo equalization in the perspective of message passing on factor graphs, we show various ways of introducing memory in the encoding process and compare the performance improvement through simulations. We also discuss an effective way of decoding such spatially coupled system by using window decoding in order to minimize the decoding latency and complexity.

We manged to show that with spatial coupling we can have it both ways; that is, have a good performance in the waterfall region while having low error floors.

Acknowledgements

I would like to express my gratitude to my supervisor Michael Lentmaier for his invaluable support throughout the period of working with this thesis. My thanks also to Muhammad Umar for his help during the project. Many thanks also to Wei Zhou for his support in some parts of the work.

Finally I would like to thank my wife Victoria, my son Samweli and my whole family for their tremendous support and patience throughout my studies.

Popular Science Summary

It is hard to imagine modern life without digital communications. It is estimated that nearly 5 billion people use mobile phones worldwide. A mobile phone nowadays is no longer a device to just make calls and send short messages but it is increasingly becoming integrated into the internet offering a range of remote services and interactive applications. Apart from mobile phones we have computers for personal and business uses most of which are now interconnected in the internet in addition to television and radio broadcasting. In addition to aspects that every one interacts directly in daily lives we also have a wide range of other areas such as underwater communications, sensor networks, industrial automation and so on.

To transfer information from one point to another most digital communication systems has to use some medium to transmit some form of a wave. The information is represented by zeros and ones which are grouped to form symbols transmitted over a range of time and frequency. As the demand for more speed and volume of data to be transferred increases, a strain is placed on the communication system to use the smallest time for each symbol and more range of frequencies (bandwidth) to meet these demands. There however some impairments which makes it harder to reach the desired speed. If the medium is wireless, for example, the transmitted signal may take several paths to reach the receiver which may result into symbols sent at one time interval to interfere with symbols sent at later times a phenomenon called inter-symbol interference (ISI). Inter-symbol interference also happens with wired medium if there is limited bandwidth or distortions in the frequency range of the signal.

To mitigate the effect of ISI it is customary for the receiver to employ some form of equalization by which the effect of other symbols is taken care of in determining the transmitted symbols. The data is also protected by some form of error correcting codes where some controlled redundancy is added to help the receiver correct some of the errors. To get good results it is important these two tasks be performed together instead of treating them separately. One way to do this with reasonable complexity is to use an iterative receiver in which the equalizer and error control decoder exchange some information in a number of cycles. This scheme is called turbo equalization.

The design of a turbo equalization is usually done with the trade-off between having good performance at low signal-to-noise-ratio (SNR) commonly called as

the waterfall or having good error floors at higher SNR. This thesis investigates how we can use spatial coupling, where blocks of codewords over time are interlinked to form a chain, can be applied to turbo equalization. It is demonstrated that with spatial coupling we can go around this trade-off and obtain good performance in the waterfall while having good error floors.

Table of Contents

1 Introduction **1**
- 1.1 Project Goal . . . 2
- 1.2 Related Work . . . 2
- 1.3 Outline of Report . . . 2

2 Background **5**
- 2.1 Introduction . . . 5
- 2.2 Equalizers . . . 6
- 2.3 Codes on Graphs . . . 9
- 2.4 EXIT Charts . . . 17
- 2.5 Capacity of an ISI channel . . . 18

3 Turbo Equalization **21**
- 3.1 Turbo equalization with convolutional code . . . 22
- 3.2 Turbo equalization with LDPC code: Weak code versus strong code 28

4 Spatial coupling **33**
- 4.1 Spatial Coupling between the inner and outer code for SCC codes . . 33
- 4.2 Coupling between interleaver and channel . . . 35
- 4.3 Turbo equalization with spatial coupling . . . 37
- 4.4 Comparison of spatial coupling with irregular LDPC codes . . . 42

5 Conclusion and future work **45**
- 5.1 Conclusions . . . 45
- 5.2 Future work . . . 46

References **47**

List of Figures

2.1 Transmitter and channel 5
2.2 Discrete time model of an ISI channel h_1 6
2.3 Tap delay line representation of the channel II 7
2.4 Trellis of channel II 8
2.5 Factor graph of an LDPC code 10
2.6 Protograph of a regular (3,6) code 12
2.7 Recursive systematic Convolutional Encoder for (1,5/7) code 13
2.8 Factor graph (a) compact representation (b) of a rate 1/2 systematic convolutional code 14
2.9 Recursive computation of $\alpha(s_n)$ and $\beta(s_n)$ for the zero state solid lines represent a 0 input while dotted line a 1 15
2.10 Serial concatenation code of two systematic rate 1/2 convolutional codes to form a rate 1/4 code 16
2.11 Factor graph (a) and compact graph representation (b) of a SCC code with N trellis sections in the outer code 16
2.12 Puncturing scheme for SCC 17
2.13 Simulation scheme for the equalizer EXIT curve 18
2.14 EXIT chart and decoding trajectory of MAP equalizer and (7,5) systematic convolutional code at 4dB 19
2.15 Capacity for IUD input for ISI channels 20

3.1 Turbo equalization scheme 21
3.2 Block diagram of MAP equalizer 22
3.3 Graphical representation of an APP equalizer with rate $1/2$ convolutional code 23
3.4 Results of turbo equalization with MAP equalizer and convolutional code, with interleaver length of 10000 24
3.5 Results of turbo equalization with MMSE equalizer and convolutional code and interleaver length of 10000 26
3.6 EXIT chart of MAP equalizer and LMMSE equalizer showing the point where the tunnel has significant opening 27
3.7 Comparison of the performance of MAP and LMMSE for interleaver length of 10^5 and 20 iterations demonstrating the effect of a significant open tunnel 28

3.8 Factor graph of LDPC code in turbo equalization. Punctured symbols are shown with no connection to the channel graph 29
3.9 Simulation strategy of arbitrary code 29
3.10 Comparison of turbo equalization with LDPC code and a convolutional code demonstrating the trade-off of choosing a weak versus a strong code 30
3.11 EXIT chart for (1,5/7) convolutional code, a regular (3,6) and 5G LDPC code with MAP equalizer 31

4.1 Spatially coupled SCC with $m = 1$ 33
4.2 Compact graph representation for SC-SCC 34
4.3 Coupling of a punctured SCC code 34
4.4 Factor graph representation of coupling between interleaver and channel 35
4.5 Compact graph representation of coupling between interleaver and channel for BPSK modulation 35
4.6 Decoding of a spatially coupled SCC system the effect of known bits with for coupling at channel interleaver with a regular (3,6) LDPC code as the component code 37
4.7 Window decoder for window size 5, decoding block 2. At this stage block 2 to 5 have some partial information obtained during decoding of block 1 37
4.8 Simulation results for coupling between interleaver and channel using a convolutional code for interleaver length of 10000 38
4.9 Results of coupling in turbo equalization with LDPC code illustrating the effect of coupling at channel and using a coupled code 39
4.10 Compact graph representation of turbo equalization with SC-SCC (a) SCC coupling at channel (b) 40
4.11 Simulation results of different ways of coupling a turbo equalization system with SCC code 40
4.12 Protograph of a coupled regular (3,6) LDPC code 41
4.13 Graphical representation of turbo equalization with a coupled LDPC code 42
4.14 Illustration of window decoding of a coupled LDPC code for turbo equalization with a coupled LDPC code for channel 1 43

List of Tables

2.1 Minimum SNR for reliable transmission for some ISI channels 20

4.1 Comparison of spatial coupling with optimized irregular LDPC codes showing SNR at which the BER falls below 10^{-5} 43

Chapter 1

Introduction

Digital communication has been an integral part of modern life. It ranges from television, radio, big data and the rapidly growing mobile communication. The volume of data to be transferred and the speed at which it has to be accessed keeps increasing as more and more parts of daily life and business gets connected in the communication networks. This makes the design of efficient digital communication systems an important aspect to make its possible to meet these requirements. Error control coding is a vital aspect in the design of these efficient systems to make it possible to communicate at higher rates with low energy requirements.

With more demands on the data rate it is common for most practical systems to suffer from inter-symbol interference(ISI) where a symbol transmitted at one symbol time spreads over to other symbols thus compromising the integrity of the received data. This spread can be caused by multipath channels commonly encountered in wireless systems where the signal takes several paths with different lengths thus arriving at different times. It is also caused by limited bandwidth for both wireless and wired systems, causing the waveform to be smeared and spread out. One common way to mitigate the effect of ISI is to use an equalizer at the receiver. An optimal equalizer is the maximum *a posterior* (MAP) equalizer. It also common to use linear equalizers such as linear minimum mean squared error (LMMSE) or zero forcing (ZF) equalizers since the MAP equalizer has a complexity order that grows exponentially with the modulation alphabet and the channel memory.

The data is also usually protected by error control coding where some structured redundancy is introduced such that the receiver can correct some errors. The two tasks of equalization and decoding are commonly performed separately causing significantly poorer performance compared to what could be achieved by an optimal joint receiver that takes into account both the code constraints and the effect of the ISI channel. The complexity of the optimal system, however, is prohibitively high for most practical systems. A better approach is to use the turbo principle where the two tasks are performed iteratively with the equalizer and decoder exchanging information for a number of cycles. This approach shows significant improvement with reasonable complexity.

The design of an iterative decoding system is usually done with a trade-off in performance in two regions; the waterfall region where the bit error rate (BER) falls sharply with signal-to-noise ratio (SNR) and the error floor region is higher

SNR where there is little decrease in the BER rate with SNR. The trade off is such that using a weak code usually results in good performance by having the waterfall at low SNR but the error floor is high whereas using a strong code gives a low error floor but the waterfall occurs at a higher SNR. Recent research in spatially coupled codes has shown that spatially coupled codes can have a belief propagation (BP) threshold asymptotically approaching that of a MAP decoder without changing the minimum distance of the uncoupled code. This motivates the application of the idea of coupling in turbo equalization.

1.1 Project Goal

In this thesis turbo equalization using the MAP and LMMSE equalizers is investigated for various codes. The performance of a weak code represented by a convolutional code with short memory is compared to that of serially concatenated codes, regular LDPC code and 5G LDPC code showing various aspects affecting the performance of a turbo equalization and the trade-offs involved in the choice of codes.

First the performance of MAP and LMMSE are compared using the same component code. Secondly the performance of a weak convolutional code is compared to that of stronger codes by simulation and extrinsic information transfer (EXIT) charts analysis. Then spatial coupling is discussed showing different ways it can be introduced and how it can change the performance of different codes. The performance of spatial coupling is then compared to that of using optimized irregular codes.

1.2 Related Work

The thesis uses concepts in two main works. The first one is on turbo equalization by Tüchler *et al* in [20] where the concept of turbo equalization is presented showing how performance improves with iterations for MAP and LMMSE equalizers using a classical convolutinal code. The second one is on spatially coupled codes by Moloudi [15] where spatial coupling of of turbo like codes especially the serially concatenated codes is showing we can get better BP threshold with a spatially coupled serially concatenated system.

1.3 Outline of Report

This thesis is organized into five chapters. Chapter 1 introduces the thesis motivation and goals. Chapter 2 introduces the theoretical background related to turbo equalization. It describes different types of equalizers and codes in the framework of message passing on factor graphs. It also introduces EXIT charts as a tool for design and analysis of iterative decoding systems.

Chapter 3 starts by analyzing turbo equalization using different equalizers and different codes based on previous research. Furthermore it compares the performance of a classical convolutional code to more powerful codes like the 5G

LDPC codes showing the trade-offs which have to be made in choice of component codes.

Chapter 4 introduces the concept of spatial coupling and in particular various ways it can be applied to turbo equalization using different codes. Here we show how spatial coupling can change the paradigm of design of iterative systems by opening the possibility of having both a strong code with low error floor and good waterfall performance. Spatial coupling is also compared to the solution of using optimized irregular codes. Finally, conclusions and future work are discussed in Chapter 5.

Chapter 2
Background

2.1 Introduction

Most practical communication systems suffer from intersymbol interference (ISI) which is usually a result of a multipath channel. One way to mitigate this effect is to use an equalizer at the receiver. Systems also employ error correcting codes whereby some controlled redundancy is introduced in order to correct some errors. Traditionally these two tasks are performed separately resulting into poor perfomance in bit error rate (BER) far from the optimal scheme which uses a joint decoder which takes into account both the effect of the channel and the code constraints. Such a scheme, however, has a tremendously high complexity making it impractical for most systems. A better approach is to perform the two tasks iteratively where the equalizer and decoder exchange soft information for a number of cycles. This leads to a great improvement in BER while having lower complexity than the optimal scheme. Figure 2.2 shows the transmitter and channel model of the system. The encoder accepts N information bits $\mathbf{u}$ and produces N/R code

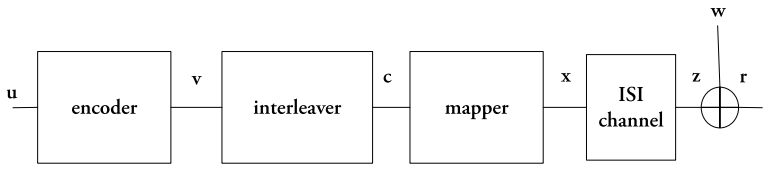

Figure 2.1: Transmitter and channel

bits $\mathbf{v}$ (where R is the rate of the code and in this thesis we use $R = 1/2$) which are interleaved and q bits are mapped to symbols $\mathbf{x}$ according to the modulation scheme, where q is the modulation order. The received signal $\mathbf{r}$ is a sum of $\mathbf{z}$ (which is the convolution of $\mathbf{x}$ with the channel impulse response $\mathbf{h}$), and white Gaussian noise $\mathbf{w}$. The mathematical model is given in (2.1)

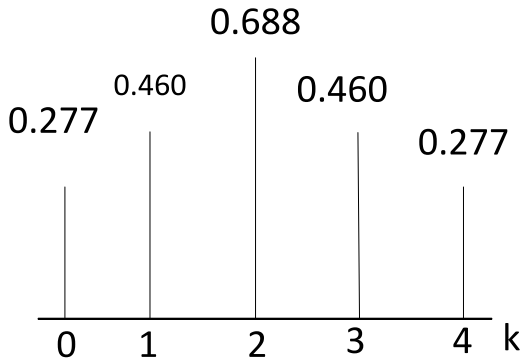

Figure 2.2: Discrete time model of an ISI channel h_1

$$\begin{aligned} \mathbf{r} &= \mathbf{z} + \mathbf{w} \\ \mathbf{r} &= \mathbf{h} * \mathbf{x} + \mathbf{w} \\ \text{i.e} \qquad r_k &= \sum_{l=0}^{l=M} h_l x_{k-l} + w_k, \end{aligned} \tag{2.1}$$

where M is the channel memory. The channels used in this thesis are from [16] $h_I = \begin{bmatrix} 0.277 & 0.46 & 0.688 & 0.46 & 0.277 \end{bmatrix}$, $h_{II} = \begin{bmatrix} 0.407 & 0.815 & 0.407 \end{bmatrix}$[20] and $h_{III} = \begin{bmatrix} \sqrt{0.45} & \sqrt{0.25} & \sqrt{0.15} & \sqrt{0.1} & \sqrt{0.05} \end{bmatrix}$. Most of the results are base on channel I. The SNR is calculated as the ratio of the received average energy per information bit (E_b) to the noise spectral density N_0. If E_s is the average energy per transmitted symbol x_n then the average energy E_z per received symbol z_n is equal to $E_z = |\mathbf{h}|^2 E_s$ where $|.|$ is the Frobenius norm. The SNR is then given by

$$\frac{E_b}{N_0} = \frac{1}{R}\frac{E_z}{N_0} = |\mathbf{h}|^2 \frac{1}{R}\frac{E_s}{N_0}.$$

2.2 Equalizers

In a separate equalization and decoding scheme the equalizer produces an estimate of the sequence of symbols $\mathbf{x}$ based on the observations $\mathbf{r}$. Different equalizers differ in the way they compute the estimate. An optimal estimate is produced by the maximum a posterior probability (MAP) equalizer which makes use of the observed symbol and the prior probabilities of the bits. This, however, has complexity order 2^{qM} which makes it unattractive for higher modulation orders and for channels with longer memory. This motivates the use of linear equalizers such as zero forcing equalizer and linear minimum mean squared error (LMMSE). The MAP equalizer is discussed first followed by the MMSE equalizer both for binary phase shift keying (BPSK) modulation.

2.2.1 Bit-wise MAP Equalization

The bitwise MAP equalizer estimates the most probable bit by taking into account the model, channel observation and *a priori* distribution of the bits. To do this it essentially computes the *a posteriori* probability of the two possible values of a bit (with the convention $0 \rightarrow +1$, $1 \rightarrow -1$) and selects the one with the highest probability. The a posterior probability for BPSK modulation is computed as follows.

$$\begin{aligned} P(x_n = x|\mathbf{r}) &= \frac{P(x_n = x, \mathbf{r})}{P(\mathbf{r})} \\ &= \frac{\sum_{\mathbf{x}:x_n=x} P(\mathbf{x}, \mathbf{r})}{P(\mathbf{r})} \\ &= \frac{\sum_{\mathbf{x_n}:x_n=+1} P(\mathbf{r}|\mathbf{x})P(\mathbf{x})}{P(\mathbf{r_n})} \\ &= \frac{\sum_{\mathbf{x_n}:x_n=+1} P(\mathbf{r}|\mathbf{x}) \prod_{k=1}^{N} P(x_k)}{P(\mathbf{r_n})}, \end{aligned}$$

where the last step assumes the symbols are independent of each other hence the probability of the vector $\mathbf{x}$ is given by the product of it components. Since the bits are a result of an encoding, they generally have some dependency which decays as the bits become far apart. The interleaver permutes these bits in pseudo-random manner to approximate the independence assumption which makes it a key component in an a system employing a turbo scheme.

There are various ways of generating interleaver such as matrix interleaver where bits are written in rows and read in columns. Another common class of interleavers are the so called S-random interleavers [7], [5] where a random bit is chosen such that any block of S bits which where neighbors in the original sequence has to be apart for a distance of more than S in the interleaved sequence. Simulation results show that S random interleavers perform better than the matrix interleavers hence throughout the thesis we use S random interleavers as described in [5].

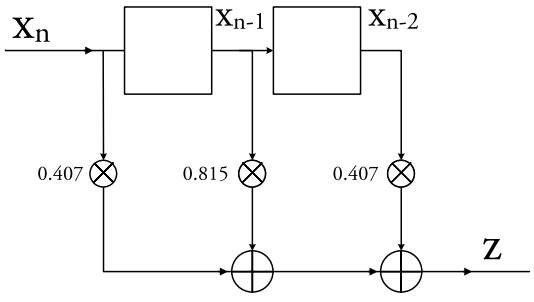

Figure 2.3: Tap delay line representation of the channel II

The channel can be represented as a tapped delay line as shown in Figure 2.3. Using this model we can construct a trellis for the channel and the computation of

the posterior probability can be effectively done on a trellis (which is elaborated in detail in Subsection 2.3.2. Figure 2.4 shows the trellis sections of Channel II. The trellis has to be initiated for the M sections assuming the input to the channel is zero for $n < 1$.

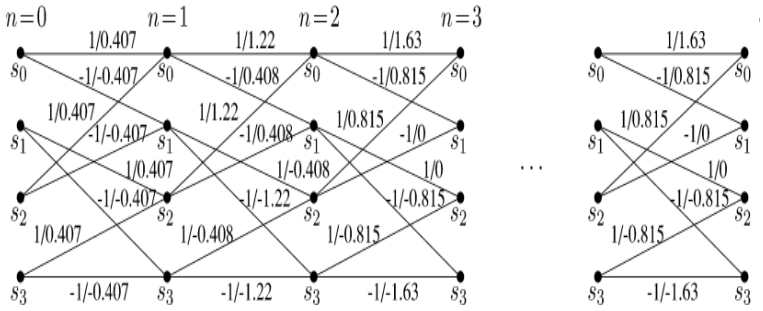

Figure 2.4: Trellis of channel II

2.2.2 Linear MMSE Equalization

Linear minimum mean squared error equalizer gives a linear estimate of the transmitted sequence which minimizes the expected squared error between the estimate and actual symbols. With the model in equation (2.1) a sequence of length N symbols is given by

$$\mathbf{r} = \mathbf{H}\mathbf{x} + \mathbf{w} \tag{2.2}$$

where $\mathbf{H}$ is an $N \times (N + M - 1)$ channel convolution matrix

$$\mathbf{H} = \begin{bmatrix} h_M & h_{M-1} & \dots & h_0 & 0 & \dots & 0 \\ 0 & h_M & h_{M-1} & \dots & h_0 & \dots & 0 \\ & \ddots & & & & & \\ 0 & \dots & 0 & h_M & h_{M-1} & \dots & h_0 \end{bmatrix}$$

and $\mathbf{x}$ represents the transmitted sequence as a vector of length $(N+M-1)$ vector $\mathbf{x} = \begin{bmatrix} x_{-M+1} & \dots & x_1 & x_2 & \dots & x_N \end{bmatrix}^T$ (It is assumed that $x_n = 0$ for $n < 1$) and $\mathbf{w}$ is length N vector $\mathbf{w} = \begin{bmatrix} w_1 & w_2 & \dots & w_N \end{bmatrix}^T$ of independent Gaussian random variables. With BPSK the transmitted signals are real taking values in S={1,-1} and the optimal estimate $\hat{x}_n$ is given by the linear filter operation

$$\hat{x}_n - E\{x_n\} = \mathbf{a}_n^T(\mathbf{r} - E\{\mathbf{r}\})$$

The optimal coefficients are given by [10]

$$\mathbf{a}_n = Cov(\mathbf{r}, \mathbf{r})^{-1} Cov(x_n, \mathbf{r})$$

where E{X} represents the expectation of X and $Cov(X, Y)$ is the covariance between X and Y and is given by

$$Cov(X, Y) = E\{XY\} - E\{X\}E\{Y\}.$$

With the model in (2.2) we have

$$\begin{aligned} E\{\mathbf{r}\} &= \mathbf{H}E\{\mathbf{x}\} \\ Cov(x_n, \mathbf{r}) &= Var(x_n)\mathbf{h_n} \\ \mathbf{\Sigma}_n = Cov(\mathbf{r}, \mathbf{r}) &= \sigma_w^2\mathbf{I} + \mathbf{H}Cov(\mathbf{x}, \mathbf{x})\mathbf{H}^T, \end{aligned}$$

where $\mathbf{h}_n$ is the n^{th} column of $\mathbf{H}$ and $Var(x_n)$ is the variance of the n^{th} symbol. Assuming the bits mapped to the symbols are independent (as discussed in Subsection 2.2.1), the symbols are then independent of each other hence the covariance matrix $Cov(\mathbf{x}, \mathbf{x})$ has zeros in all its off-diagonal elements and contains the variances of the symbols along the diagonal. If the symbols are equally likely the mean and variances of each symbol are 0 and 1 respectively, thus $Cov(\mathbf{x}, \mathbf{x})$ is an $N \times N$ identity matrix. The estimate of the n^{th} symbol is then given by

$$x_n = \left(\sigma_w^2\mathbf{I} + \mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{h}_n\Big(\mathbf{r} - E\{\mathbf{r}\}\Big)$$

2.3 Codes on Graphs

Factor graphs represents relationships between functions and variables. They can be used to model many algorithms used in various fields [11]. The decoding process of the codes used and the turbo equalization scheme are also instances of message passing in a graph, the messages that are passed around are probabilities or beliefs based on some observations hence the term belief propagation. Low density parity check codes are introduced first followed by convolutional codes and serially concatenated codes.

2.3.1 Low Density Parity Check codes

Low density parity check codes introduced by Gallager in [8] are codes characterized by having a sparse parity check matrix. An (n, k) code, where k is the number of information bits and n is the number of code-bits, has an $(n - k)$ by n parity check matrix $\mathbf{H}$. Every valid code-word $\mathbf{v}$ is in the null space of $\mathbf{H}$, that is, it must satisfy the relationship $\mathbf{H}\mathbf{v}^T = \mathbf{0}$. For example for the matrix given by 2.3 the relationship can be factored as

$$I(v_1 + v_2 + v_4 = 0) \cdot I(v_3 + v_4 + v_6 = 0) \cdot I(v_4 + v_6 + v7 = 0) = 1$$

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \tag{2.3}$$

where $I(\cdot)$ is an indicator function. This can be interpreted as each check equation (a row of $\mathbf{H}$) has to be satisfied. This factorization is represented by a bipartite graph where one group of nodes represents the symbols as columns of $\mathbf{H}$ (named

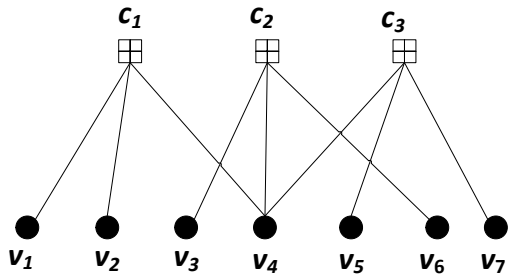

Figure 2.5: Factor graph of an LDPC code

as variable nodes) and the other group represents check equations (named check nodes) as in Figure 2.5. There is a link from check node i to variable node j if theentry (i, j) in $\mathbf{H}$ is non-zero.

The bit wise MAP decoding for symbol j in thecode-word $\mathbf{v}$ can be interpreted as the computation of a posterior probability of all codewords with symbol j fixed to either $+1$ or -1 given the observation $\mathbf{r}$ and *a priori* distribution and then choosing the symbol which maximizes this probability. This marginalization has to be done for every codesymbol. It can be shown that this marginalization can be done via message passing in a factor graph [17]. The message passed from a check node c_i to a variable node v_j is the probability that the variable node v_j takes a value in {0,1} based on the information all other variable nodes connected to the check node. In other words c_i sends the probability that it sums up to zero if the variable node v_j is fixed to a given value. Therefore there are two messages one for each possible value of v_j. For example the message from c_1 to v_1 indicating that v_1 takes the value 0, $\mu_{c_1,v_1}(0)$ is the sum of probabilities two events: both v_2 and v_4 are 0 or both v_2 and v_4 are 1, that is

$$\begin{aligned}
\mu_{c_1,v_1}(0) &= P(v_1 = 0|\text{messages from } v_2 \text{ and } v_4) \\
&= \sum_{v_2+v_4=0} \prod_{j=2,4} \mu_{v_j,c_1}(v_j) \\
&= \mu_{v_2,c_1}(0) \cdot \mu_{v_4,c_1}(0) + \mu_{v_2,c_1}(1) \cdot \mu_{v_4,c_1}(1)
\end{aligned}$$

Similary

$$\mu_{c_1,v_1}(1) = \mu_{v_2,c_1}(0) \cdot \mu_{v_4,c_1}(1) + \mu_{v_2,c_1}(1) \cdot \mu_{v_4,c_1}(0)$$

Instead of sending two distinct messages, a ratio $r(c_1, v_1)$ is formed.

$$\begin{aligned} r(c_1, v_1) &= \frac{\mu_{c_1,v_1}(0)}{\mu_{c_1,v_1}(1)} \\ &= \frac{\mu_{v_2,c_1}(0) \cdot \mu_{v_4,c_1}(0) + \mu_{v_2,c_1}(1) \cdot \mu_{v_4,c_1}(1)}{\mu_{v_2,c_1}(0) \cdot \mu_{v_4,c_1}(1) + \mu_{v_2,c_1}(1) \cdot \mu_{v_4,c_1}(0)} \\ &= \frac{1 + r(v_2, c_1) \cdot r(v_4, c_1)}{r(v_2, c_1) + r(v_4, c_1)} \end{aligned}$$

Applying some steps of algebraic manipulations we can show that

$$\frac{r(c_1, v_1) - 1}{r(c_1, v_1) + 1} = \frac{\Big(r(v_2, c_1) - 1\Big) \cdot \Big(r(v_4, c_1) - 1\Big)}{\Big(r(v_2, c_1) + 1\Big) \cdot \Big(r(v_4, c_1) + 1\Big)}$$

It customary to work with messages in the log domain. Hence defining the log-likelihood ratio as the ratio of the two probabilities we then have

$$\begin{aligned} L &= \frac{P(v_j = 0)}{P(v_j = 1)} \\ L &= \ln(r) \\ \text{which implies} \quad \frac{r-1}{r+1} &= \tanh(L/2) \end{aligned}$$

Therefore

$$\begin{aligned} \tanh\Big(\frac{1}{2}L(c_1, v_1)\Big) &= \tanh\Big(\frac{1}{2}L(v_2, c_1)\Big) \cdot \tanh\Big(\frac{1}{2}L(v_4, c_1)\Big) \\ L(c_1, v_1) &= 2\tanh^{-1}\left(\tanh\Big(\frac{1}{2}L(v_2, c_1)\Big) \cdot \tanh\Big(\frac{1}{2}L(v_4, c_1)\Big)\right). \end{aligned}$$

In general it can be shown [17] that the messages passed from c_i to an edge e_k is given by

$$L_{c_i}(e_k) = 2\tanh^{-1}\left(\prod_{k' \neq k} \tanh\left(\frac{1}{2}L_v(e_{k'})\right)\right).$$

One the other hand the message passed from a variable node to a check node is the probability that the variable node takes a value in {0,1} based on the information from the channel from the other check nodes apart from the one being updated. The probability that a variable node takes a value of 0, for example, based on the messages of the check nodes connected to it is given by the product of the probability of each of the received message referring to the variable node as zero. Therefore the messages from v_4 to c_1 are given by

$$\begin{aligned} \mu_{v_4,c_1}(0) &= \mu_{c2,v_4}(0) \cdot \mu_{c3,v_4}(0) \cdot \mu_{\text{ch},v_4}(0) \\ \mu_{v_4,c_1}(1) &= \mu_{c2,v_4}(1) \cdot \mu_{c3,v_4}(1) \cdot \mu_{\text{ch},v_4}(1) \end{aligned}$$

The logarithm of the ratio of the two messages is then given by

$$L(v_4, c_1) = L_{\text{ch}}(v_4) + L(c_2, v_4) + L(c_3, v_4)$$

In general the message passed from a variable node j to an edge e_k is given by the sum of the incoming messages from all edges except e_k. That is

$$L_{v_j}(e_k) = L_{\text{ch}}(v_j) + \sum_{k' \neq k} L_c(e_{k'}).$$

The order of message passing can be chosen in various ways. One common approach is to update all the variable nodes while freezing the check nodes followed by updating all check nodes and repeating the process for some specified number of cycles or terminate if some criteria (for example when the variable nodes correspond to a valid codeword) is satisfied.

LDPC codes can be constructed from protographs [13]. A protograph consists of a small number of check nodes and variable nodes representing the structure of the LDPC code. For example Figure 2.6 shows a protograph of a regular (3,6) code represented by the base matrix B given by

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

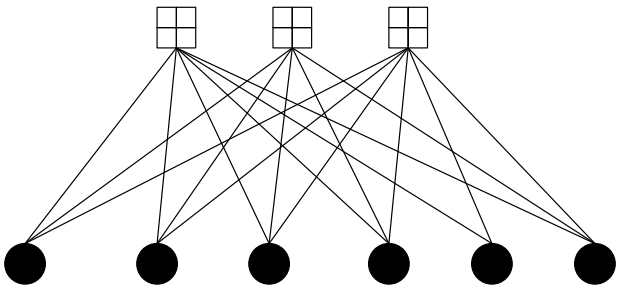

Figure 2.6: Protograph of a regular (3,6) code

The parity check matrix is obtained by an uplifting procedure [13] where the non zero entries of the base matrix are replaced with rotated identity matrices of sizes equal to the uplifting factor while the zero entries are replaced by zero matrices of the same size.

2.3.2 Convolutional Codes

Convolutional codes can be characterised by their property whereby the current output bits depends not only on the current bit(s) but also on several past bits. The convolutional code considered are recursive systematic with rate $1/2$ represented by a sequential circuit as shown in Figure 2.12 in a controller canonical form. The feed-forward and feedback polynomials are $1+D^2$ and $1+D+D^2$ respectively which

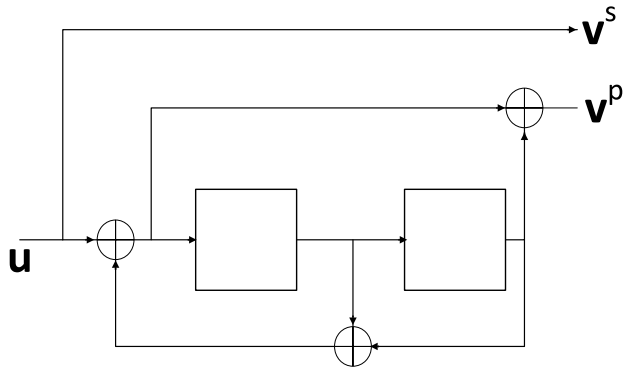

Figure 2.7: Recursive systematic Convolutional Encoder for (1,5/7) code

can also be represented in octal formats [12] as 5,7 respectively. The decoding of the convolutional code is discussed with BPSK modulation and the mapping $\mathbf{v}^s \rightarrow \mathbf{x}^s, \mathbf{v}^p \rightarrow \mathbf{x}^p$. The Bit-wise MAP decoder computes the sum of posterior probability of all possible transmitted sequences which has the bit at position n fixed to one of the values in {-1,+1} and chooses the value of x_n which maximizes this probability given the channel observations $\mathbf{r}$. That is

$$\begin{aligned} \hat{x}_n &= \underset{x\in\{-1,+1\}}{\operatorname{argmax}} \sum_{\mathbf{x}:x_n=x} P\big(\mathbf{x}|\mathbf{r}\big) \\ &= \underset{x\in\{-1,+1\}}{\operatorname{argmax}} \sum_{\mathbf{x}:x_n=x} P\big(\mathbf{r}|\mathbf{x}\big)P\big(\mathbf{x}\big) \end{aligned} \tag{2.4}$$

If we assume the symbols in the sequences $\mathbf{x}$ and $\mathbf{r}$ are independent we can expand their probabilities as product of the probabilities of their components. Further more we have two types of bits the systematic bits x_n^s and parity bits x_n^p and their corresponding observations r_n^s and r_n^p. Thus (2.4) can then be expanded as

$$\hat{x}_n = \underset{x\in\{-1,+1\}}{\operatorname{argmax}} \sum_{\mathbf{x}:x_n=x} \prod_{i=1}^{N} P\big(r_i^s|x_i^s\big)P\big(r_i^p|x_i^p\big) \prod_{i=1}^{N} P\big(x_i^s\big)P\big(x_i^p\big) \tag{2.5}$$

The events are however conditioned on the state transitions which are node observable. Therefore the marginalization can be computed introducing state variables s_i.

$$\begin{aligned} \hat{x}_n = \underset{x\in\{-1,+1\}}{\operatorname{argmax}} \sum_{\mathbf{x}:x_n=x} P(s_0) \prod_{i=1}^{N} \underbrace{P\big(r_i^s|x_i^s\big)P\big(r_i^p|x_i^p\big)}_{\text{channel}} \underbrace{\prod_{i=1}^{N} P\big(x_i^s\big)P\big(x_i^p\big)}_{\text{prior}} \\ \cdot \underbrace{P(x_i^p, s_i|x_i^s, s_{i-1})}_{\text{valid transitions}} \end{aligned} \tag{2.6}$$

In SISO decoding the decoder does not provide the estimate of x_n but produces the probabilities instead, hence the name a posterior probability (APP) decoder. The factor graph of this marginalization is depicted in Figure 2.8 (a). The squared box represent an indicator function for valid transitions $P(x_i^p, s_i | x_i^s, s_{i-1})$. The factor graph can be represented in a compact form by grouping nodes whose distribution are similar. In this case, ignoring the edge effects (symbols at the start and end of a trellis are affected by knowledge of start and end states), we have two types of nodes: systematic and parity nodes as shown in Figure 2.8 (b). Instead of nodes passing probabilities LLRs are passed instead as defined in Subsection 2.3.1.For example the LLR for the n^{th} systematic bit is given by

$$
\begin{aligned}
L(x_n^s) =& \ln \frac{\sum\limits_{\mathbf{x}: x_n^s = +1} P(s_0) \prod\limits_{i=1}^{N} P\big(r_i^s | x_i^s\big) P\big(r_i^p | x_i^p\big) P\big(x_i^p, s_i | x_i^s, s_{i-1}\big) P\big(x_i^s\big) P\big(x_i^p\big)}{\sum\limits_{\mathbf{x}: x_n^s = -1} P(s_0) \prod\limits_{i=1}^{N} P\big(r_i^s | x_i^s\big) P\big(r_i^p | x_i^p\big) P\big(x_i^p, s_i | x_i^s, s_{i-1}\big) P\big(x_i^s\big) P\big(x_i^p\big)} \\
=& \underbrace{\ln \frac{\sum\limits_{x_n^s = +1} P(s_0) \prod\limits_{i=1}^{N} P\big(r_i^s | x_i^s\big) P\big(r_i^p | x_i^p\big) P\big(x_i^p, s_i | x_i^s, s_{i-1}\big) P\big(x_i^p\big) \prod\limits_{i \neq n} P\big(x_i^s\big)}{\sum\limits_{x_n^s = -1} P(s_0) \prod\limits_{i=1}^{N} P\big(r_i^s | x_i^s\big) P\big(r_i^p | x_i^p\big) P\big(x_i^p, s_i | x_i^s, s_{i-1}\big) P\big(x_i^p\big) \prod\limits_{i \neq n} P\big(x_i^s\big)}}_{\text{extrinsic}} \\
& + \underbrace{\ln \frac{P\big(x_i^s = +1\big)}{P\big(x_i^s = -1\big).}}_{\text{prior}}
\end{aligned}
$$

Hence the LLR is the sum of two terms; the extrinsic and *a priori* LLRs. The

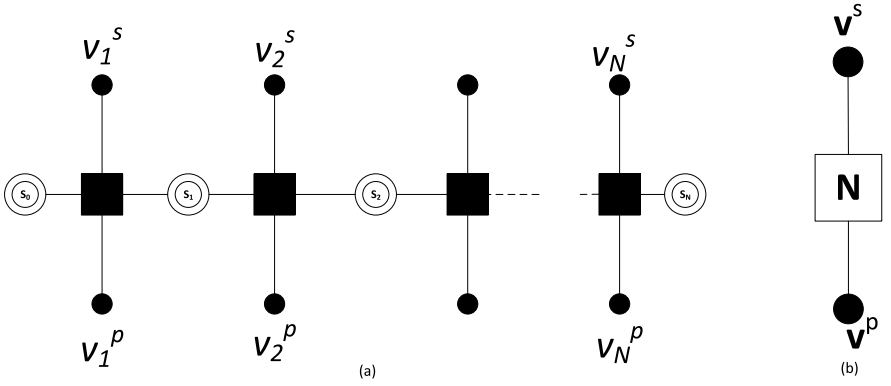

Figure 2.8: Factor graph (a) compact representation (b) of a rate 1/2 systematic convolutional code

extrinsic LLRs is the information about x_n based on channel observations and prior from other bits. The computation of the marginals can be done efficiently by using the trellis. Which divides the posterior probability $P\big(x_n | \mathbf{r}\big)$ into three parts[1]:

- transitions before time n $\alpha_{n-1}(s_{n-1})$,
- transitions after n $\beta_n(s_n)$,
- local transition $\gamma_n(s_{n-1}, s_n)$

The local evidence or branch metric $\gamma_n(s_{n-1}, s)$ is given by

$$\gamma_n(s_{n-1}, s_n) = P\big(r_n^s|x_n^s\big)P\big(r_n^p|x_n^p\big)P\big(x_n^s\big)P\big(x_n^p\big)$$

while $\alpha_n(s_n)$ and $\beta_n(s_n)$ are computed recursively as

$$\alpha_n(s_n) = \sum_{s_{n-1}, x_n^s, x_n^p} \alpha_n(s_{n-1})\gamma_n(s_{n-1}, s_n)P(x_n^p, s_{n-1}|x_n^s, s_n),$$

$$\beta(s_n) = \sum_{s_{n+1}, x_n^s, x_n^p} \beta(s_{n+1})\gamma_{n+1}(s_n, s_{n+1})P(x_{n+1}^p, s_{n+1}|x_{n+1}^s, s_n).$$

The LLR is then calculated as

$$L(x_n) = \frac{P\big(x_n = +1|\mathbf{r}\big)}{P\big(x_n = -1|\mathbf{r}\big)} = \frac{\sum\limits_{x_n=+1} \alpha_{n-1}(s_{n-1})\gamma_n(s_{n-1}, s_n)\beta_n(s_n)}{\sum\limits_{x_n=-1} \alpha_{n-1}(s_{n-1})\gamma_n(s_{n-1}, s_n)\beta_n(s_n)}$$

To avoid numerical issues the computation is done in log domain and the $\alpha(s_n)$ and $\beta(s_n)$ are normalized by their respective sums at each trellis stage as described in [1].

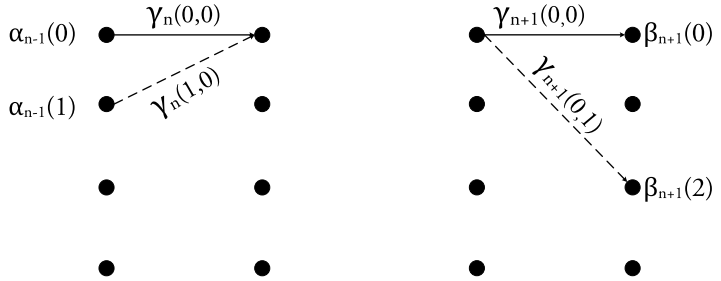

Figure 2.9: Recursive computation of $\alpha(s_n)$ and $\beta(s_n)$ for the zero state solid lines represent a 0 input while dotted line a 1

2.3.3 Serially Concatenated codes

Serial concatenation consists of two cascaded component codes, separated by an interleaver. If the codes used are recursive convolutional codes with rate 1/2 each, the scheme could be described as follows using Figure 2.10. The outer encoder accepts N information bits $\mathbf{u}$ and produces N parity bits $\mathbf{v^o}$ which are multiplexed with $\mathbf{u}$ and interleaved and act as input to the inner component code. The inner code produces the parity bits $\mathbf{v^I}$. Finally the output $(\mathbf{u}, \mathbf{v^o}, \mathbf{v^I})$ is then

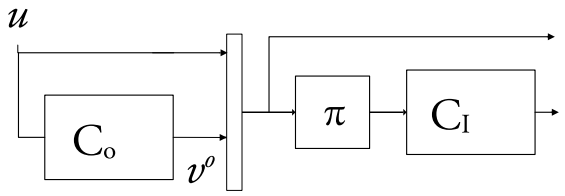

Figure 2.10: Serial concatenation code of two systematic rate 1/2 convolutional codes to form a rate 1/4 code

transmitted.The compact graph representation is shown in Figure 2.11. In this arrangement the extrinsic LLRs from the inner code supplies the prior to both the parity and systematic bits of the outer code while the extrinsic LLRs of the outer code act as prior to the systematic bits of the inner code.

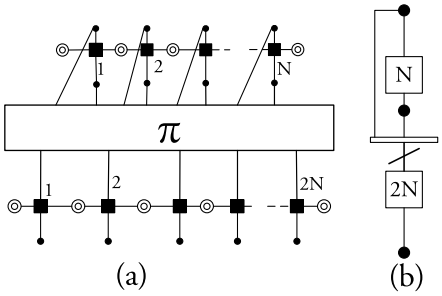

Figure 2.11: Factor graph (a) and compact graph representation (b) of a SCC code with N trellis sections in the outer code

The code can be punctured to get higher rates than $1/4$. The puncturing pattern has an effect on the performance of the code. The pattern used follows a random puncturing scheme described in [2] to obtain a rate $1/2$ code. A puncturing pattern $\mathbf{P}_i$ is represented as a vector equal to the length of the un-punctured vector of bits it punctures. A zero in the pattern means a bit in that position is punctured while a one implies it is un-punctured. The puncturing pattern is formulated for an outer code input vector of 200 bits and longer lengths are obtained by repeating the pattern. The systematic bits of the outer code $\mathbf{u}$ are punctured by a random puncturing pattern $\mathbf{P}_o$ to get d_0 bits, $\mathbf{u}_0$. The 200 parity bits are punctured by a pattern $\mathbf{P}$ which has a repeating pattern [1 0] meaning that every second parity bit is punctured remaining with 100 bits which are further puncture by a random pattern of length 100 $\mathbf{P}_1$ to produced d_1 bits ,$\mathbf{v}_1$. The 100 bits from $\mathbf{P}$ are also combined with the 200 systematic bits by a multiplexer and interleaved to act as input to the inner encoder. The 300 parity bits of the inner encoder are punctured to obtain d_2 bits $\mathbf{v}_2$. Finally $\mathbf{u}_0$,$\mathbf{v}_1$ and $\mathbf{v}_2$ are multiplexed and transmitted. The rate of the resulting code is then given by

$$R = \frac{200}{d_0 + d_1 + d_2}$$

Following the scheme in[2] the values are chosen as $d_0 = 200, d_1 = 100$ and $d_2 = 100$

to obtain $R = 1/2$ optimized for good performance in the waterfall region.

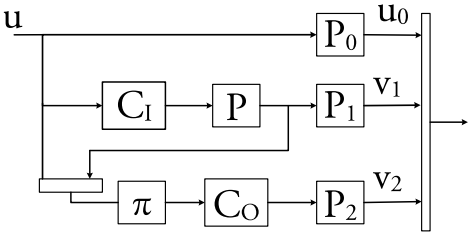

Figure 2.12: Puncturing scheme for SCC

2.4 EXIT Charts

Extrinsic information transfer (EXIT) charts introduced by ten Brink [18], are used to predict the convergence of iterative decoding systems. These characterize the statistics of the LLRs by the mutual information. Mutual information defined in (2.7) represents the amount of information about the symbols contained in the LLRs. That is, $I = 0$ information corresponds to no information(the bits are equally likely to be 0 or 1 equivalent to zero LLRs) while $I = 1$ represents perfect knowledge of bits.

$$\begin{aligned} I(X, L) &= \mathrm{E}\Big\{\log \frac{P(X, L)}{P(X)P(L)}\Big\} \\ &= H(L) - H(L/X); \end{aligned} \tag{2.7}$$

In general, the statistics of the bits differ depending on whether the bit is systematic or parity and on the position in the block.But since we are interested in the LLR reaching at the input of the equalizer and at the input of the decoder the mutual information is computed as the average of all bits which is justified by the presence of an inter-leaver between the components decoders. Following the works in [18] and [9] the mutual information is obtained by simulation using the equation

$$I(X, L) = 1 - \mathrm{E}\Big\{\log(1 + e^{-xL})\Big\} \tag{2.8}$$

The equalizer has access to the channel while the code does not. Thus the simulation for the two differ. The aprior distribution of the channel can be modelled as Gaussian distributed with variance σ_A^2 and mean $\mu_A \cdot x$ with $\mu_A = \sigma_A^2/2$. For this distribution 2.7 is represented by a function $J(\sigma_A)$ defined as

$$J(\sigma_A) = 1 - \int_{-\infty}^{\infty} \frac{e^{-\left(L - \sigma_A^2/2\right)^2/(2\sigma_a^2)}}{\sigma_A\sqrt{2\pi}} \log(1 + e^{-L}) dL$$

whose inverse can be computed numerically to obtain σ_A for known value of $I(X, L)$ since the function is monotonically increasing with σ_A. This is done by

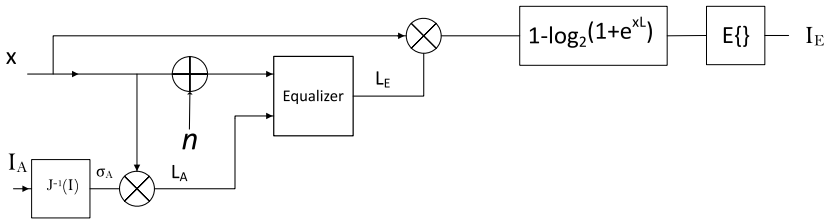

Figure 2.13: Simulation scheme for the equalizer EXIT curve

computing the J function for a range of values from 0 to an arbitrary large value such as 100 with any required precision creating a lookup table which can be used in computing the inverse. The noise is then normalized according to the SNR and rate of the code used. In this way different curves for varying SNR can be computed. The scheme is shown in Figure 2.13

For the code on the other hand we only have the *a priori* values which we expect from the channel. The EXIT curve of the code can be obtained in two ways. One by simulating the code and channel together while changing the aprior mutual information to the channel and using the extrinsic information from the channel as aprior to the code. This method however will give a limited curve for the code as it will cover only the range of values of mutual information which can be the output of the channel. Alternatively we can assume Gaussian distributed LLRs for a given mutual information. This assumption is supported by simulations where the distribution of the extrinsic LLRs from the channel plotted in an empirical histogram is not so far from the normal curve and further more the code's EXIT curves obtained in the two different ways show no noticeable deviations. The curves shown are obtained using the second method.

The behaviour of an iterative decoding system in the waterfall region can be deduced by plotting the transfer functions of the two component decoders with the second transfer function inverted. If the two curves intersect only at a point of perfect information (i.e mutual information is 1) then the system has the possibility to converge after many iterations provided the interleaver is long enough to avoid correlations before reaching the perfect knowledge state. We can also plot the evolution of the mutual information with iterations for any two components in an iterative system. Figure 2.14 shows the EXIT chart and the decoding trajectory for the turbo equalizer using MAP equalizer and the (1,5/7) convolutional code at 4 dB. The EXIT chart can also be used to predict the BER if both the *a priori* and extrinsic LLRs are assumed to be Gaussian. With this approximation it can be shown that the higher the mutual information the lower the BER rate. So decoding trajectories which stop at points of low mutual information implies a higher BER than those stopping at a higher point.

2.5 Capacity of an ISI channel

The capacity of a channel is defined as the maximum information rate between the input $\mathbf{X}$ and output $\mathbf{R}$ of a communication channel with the maximization

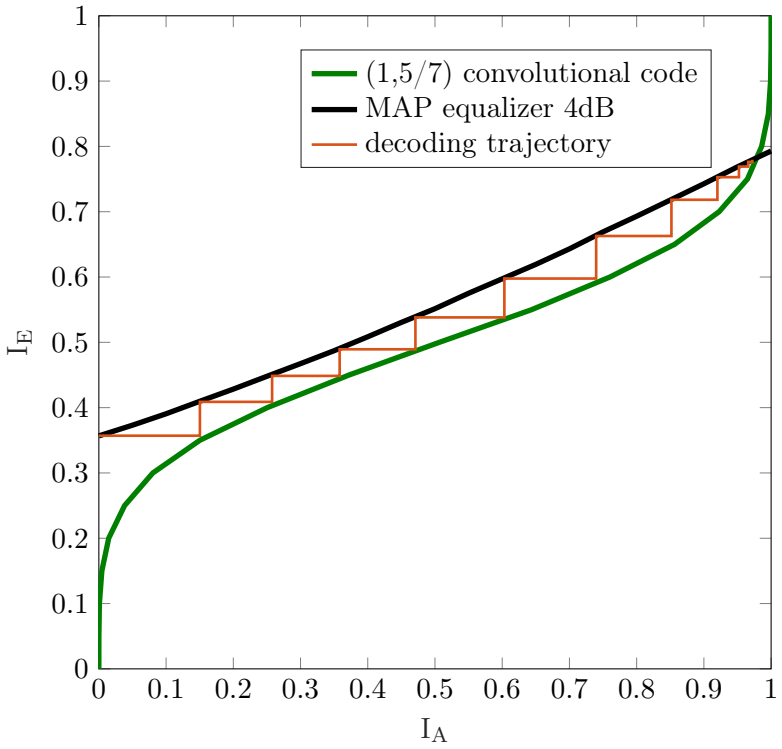

Figure 2.14: EXIT chart and decoding trajectory of MAP equalizer and (7,5) systematic convolutional code at 4dB

done over all possible distributions of the input $\mathbf{X}$. If both $\mathbf{X}$ and $\mathbf{R}$ are length L vectors then the capacity is given by

$$C = \max_{p(X)} \lim_{L\to\infty} \frac{1}{L} I(\mathbf{X};\mathbf{R})$$

where $I(\mathbf{X};\mathbf{R})$ is the mutual information between $\mathbf{X}$ and $\mathbf{R}$. Therefore to achieve capacity we need to choose an input distribution which maximizes the information rate without exceeding the power constraint. For the scope of this thesis, however we restrict ourselves to the case where the input symbols are identical and uniformly distributed. This means for the case of BPSK modulation the symbols x_n takes one of the two possible values in $\{-1,+1\}$ with probability $1/2$. We call this constrained capacity for identically and uniformly distributed input($C_{i.u.d}$). Using properties of mutual information [6] and the channel model in (2.1) $C_{i.u.d}$ for BPSK modulation is given by

$$\begin{aligned} C_{i.u.d} &= \lim_{L\to\infty} \frac{1}{L}\Big(H(\mathbf{R}) - H(\mathbf{R}|\mathbf{X})\Big) \\ &= \lim_{L\to\infty} \frac{1}{L}\Big(H(\mathbf{R}) - H(\mathbf{W})\Big) = \lim_{L\to\infty} \frac{1}{L} H(\mathbf{R}) - \log_2(2\pi e\sigma_n^2). \end{aligned}$$

where $H(\mathbf{X})$ is the entropy of $\mathbf{X}$. This capacity is then computed using a simulation method on a trellis as described in [3]. Figure 2.15 shows the plot of $C_{i.u.d}$ for the three channels used for different $\frac{Es}{N_0}$ with BPSK modulation. From the Figure we can deduce the minimum $\frac{E_s}{\sigma^2}$ and the corresponding $\frac{E_b}{N_0}$ for reliable transmission with rate 1/2 as shown in Table 2.1

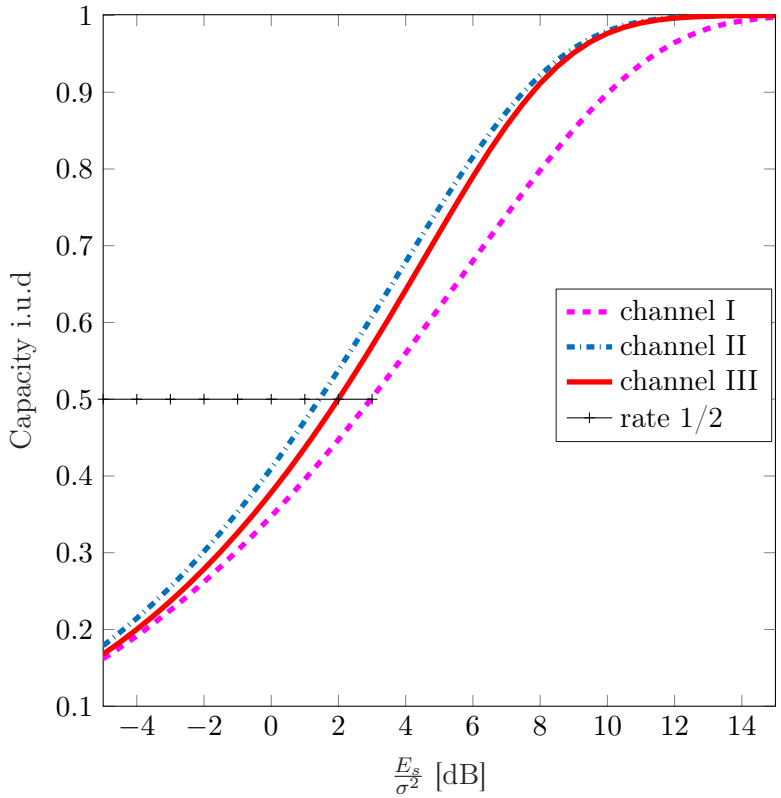

Figure 2.15: Capacity for IUD input for ISI channels

channel	Min. $\frac{Eb}{N_0}$
h_I $[0.277 \quad 0.46 \quad 0.688 \quad 0.46 \quad 0.277]$	2.95
h_{II} $[0.407 \quad 0.815 \quad 0.407]$	1.4
h_{III} $[\sqrt{0.45} \quad \sqrt{0.25} \quad \sqrt{0.15} \quad \sqrt{0.1} \quad \sqrt{0.05}]$	2

Table 2.1: Minimum SNR for reliable transmission for some ISI channels

Chapter 3

Turbo Equalization

Motivated by the success of turbo codes, turbo equalization uses the idea of exchanging soft information between the equalizer and decoder to improve the performance the receiver for an ISI channel. The code and the channel together form a system which is equivalent to a serially concatenated code discussed in Subsection 2.3.3. As in turbo codes, to get a good performance the equalizer and the code has to communicate some form of of messages showing with how much probability the bits take one of their possible values. This is achieved by exchanging extrinsic LLRs, which are has already been discussed for the decoders in 2.3.2. In this chapter we will discuss how soft information is produced for MAP and MMSE equalizers and how the performance changes using different types of codes.

The principle of turbo equalization can be described as follows: The equalizer produces extrinsic LLRs L_e^E for the bits mapped to the symbols based on the channel observations and aprior LLRs L_a^E. The extrinsic LLRs of the equalizer after de-interleaving L_a^D are used as *a priori* LLRs in the decoder marking one iteration. In the next iteration the equalizer uses the decoders extrinsic LLRs as its *a priori*. The *a priori* LLRs to the equalizer are initialized to zero during the first iteration. The two components exchange their extrinsic LLRs for a number of iterations. The arrangement is shown in Figure 3.1.

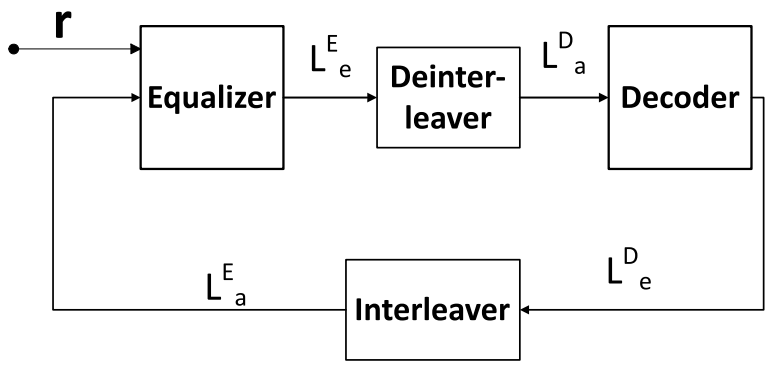

Figure 3.1: Turbo equalization scheme

3.1 Turbo equalization with convolutional code

In this setup the (1,5/7) convolutional encoder is used to illustrate the principle of turbo equalization with a relatively weak code (We can obtain a more powerful convolutional code by choosing one with more memory).

3.1.1 Turbo equalization with bit-wise MAP equalizer

Bit wise equalizer for BPSK modulation computes the posterior probability in the same fashion as a convolutional code in Subsection 2.3.2. The difference between the two is that the channel transmits its output symbols $\mathbf{z}$ but not its input symbols $\mathbf{x}$. Thus the factor graph of an APP equalizer shown in Figure 3.3 has the same structure as a rate 1/2 convolutional code but with the systematic symbols punctured. This means the marginalization can also done one a trellis using the BCJR algorithm. The branch metric for the equalizer is given by

$$\gamma_n(s_{n-1}, s_n) = P\big(r_n|z_n\big)P\big(x_n\big).$$

With the model in (2.1) and from definition of LLR we have

$$P\big(r_n|z_n\big) = \tfrac{1}{\sigma_n\sqrt{2\pi}} e^{-\frac{(r_n - z_n)^2}{2\sigma_n^2}},$$

$$P(x_n = x) = \frac{e^{xL(x_n)/2}}{e^{L(x_n)/2} + e^{-L(x_n)/2}}$$

In log domain the branch metric is then given by

$$\Gamma_n(s_{n-1}, s_n) = A + x\frac{L(x_n)}{2} - \frac{(r_n - z_n)^2}{2\sigma^2},$$

$$A = -\ln\Big(\sigma_n\sqrt{2\pi}\Big) - \ln\Big(e^{L(X_i)/2} + e^{-L(X_i)/2}\Big).$$

Since A does not depend on the sign of the symbol it can be removed as it appears on both sides in the computation of the LLRs from the equalizer. Similar to the convolutional code the extrinsic LLR L_e^E of the APP equalizer is obtained by subtracting the *a priori* LLR L_a^E from the output LLR for each symbol.

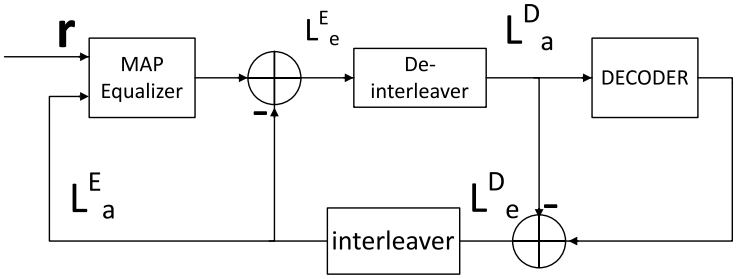

Figure 3.2: Block diagram of MAP equalizer

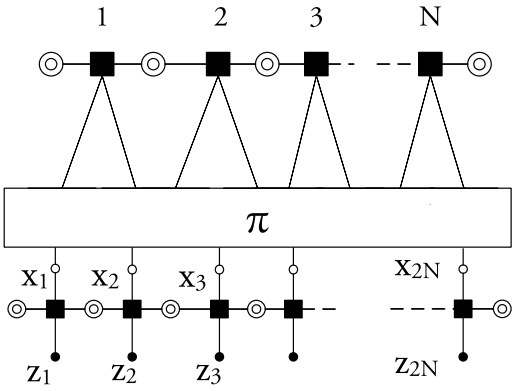

Figure 3.3: Graphical representation of an APP equalizer with rate 1/2 convolutional code

The convolutional code uses the LLRs from the equalizer as prior to both its code bits (both systematic and parity). Since there is no channel information the branch metric is then given by

$$\begin{aligned}\gamma_n(s_{n-1}, s_n) &= P\big(x_i^s\big)P\big(x_i^p\big) \\ &= \frac{e^{x_n^s L(x_n^s)/2}}{e^{L(x_n^s)/2} + e^{-L(x_n^s)/2}} \cdot \frac{e^{x_n^p L(x_n^p)/2}}{e^{L(x_n^p)/2} + e^{-L(x_n^p)/2}}\end{aligned}$$

which in log domain (after removing constants as in the equalizer) can be expressed as

$$\Gamma_n(s_{n-1}, s_n) = \frac{1}{2}\Big(x_n^s L(x_n^s) + x_n^p L(x_n^p)\Big).$$

This scheme can be viewed in the frame of message passing with the graph being the combined graph of the code and the equalizer with the interleaver in between as depicted in Figure 3.3.

Figure 3.4 shows the simulation results for both separate equalization and decoding and turbo equalization with different number of iterations for interleaver length of 10000. It can be seen that SISO equalization shows improvement over hard decision separate equalization and decoding even with one iteration. The performance improves with iterations such that with 10 iterations the performance approaches that of a code in an ISI free channel for SNR above 4 dB. It can be observed that the gain diminishes with more iterations. For example the gain from 1 to 2 iterations for BER of 10^{-5} is about 4 dB but we see only 1.5 dB gain when we increase the iterations from 2 to 3 despite not reaching the ISI limit. This gain reduction with increasing iterations is due to the fact that the extrinsic LLRs from the code and channel become more and more correlated with increasing iterations due to cycles in the interleaver and the overall graph. These cycles means that at some point the variables receive information which initially originated from themselves. So increasing the interleaver length will provide more gain for each iteration and thus fewer iterations might be needed to reach the limit imposed by the code or the threshold of the system if it exists. With this convolutional

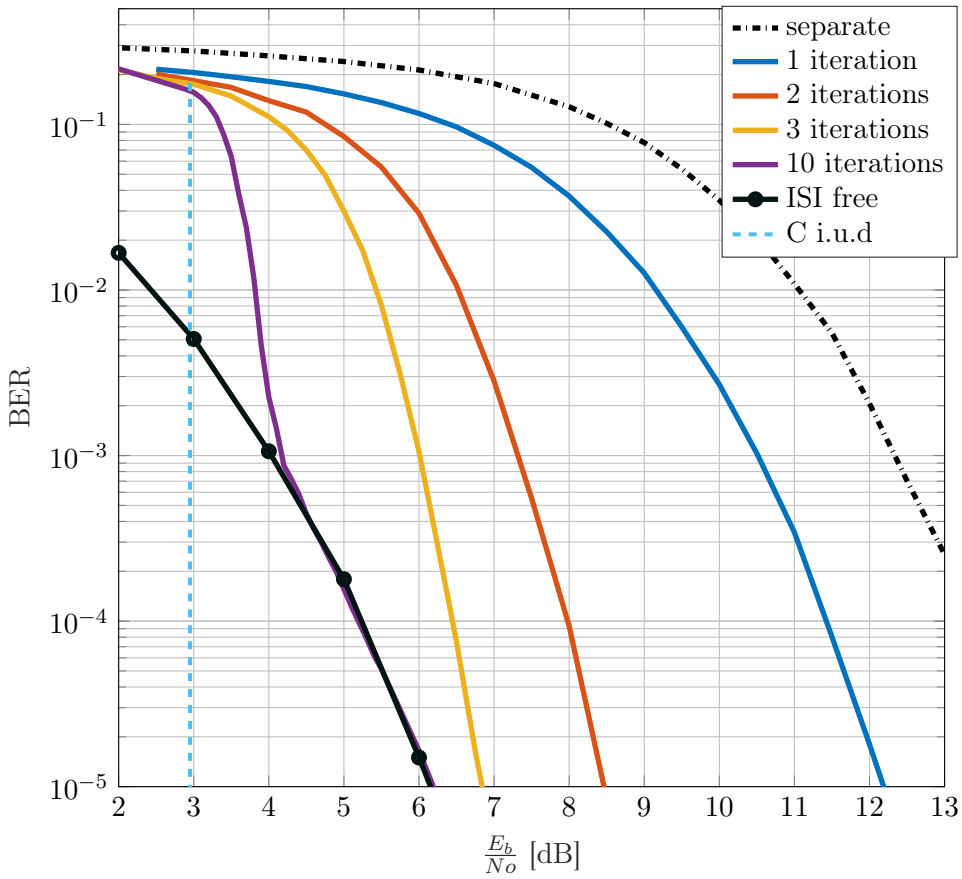

Figure 3.4: Results of turbo equalization with MAP equalizer and convolutional code, with interleaver length of 10000

code, however, we cannot observe a clear threshold. These observations can be explained by the shape of the EXIT curves of the code and channel. As it can be seen in Figure 2.14 the code's curve intersects with the equalizer for most range of SNRs thus limiting the BER to what can be predicted by the intersection point.

3.1.2 Turbo equalization with linear MMSE Equalizer

To provide soft output for a linear MMSE equalizer we can use the statistics of the estimation error for each symbol. The derivation in the thesis follows closely the work in [20]. Instead of using the whole sequence $\mathbf{r}$ to estimate one symbol x_n we can use a window of $W + M$ symbols $\mathbf{r_n}$ from r_{n-N_2-M} to r_{n+N_1} where M is the channel memory and $W = N_1 + N_2 + 1$. The submatrix $\mathbf{H}_n$ has the same structure as $\mathbf{H}$ in Subsection 2.2.2 but has dimensions $W \times (W + M)$ instead. This reduces the complexity while showing no significant reduction in performance degradation as long as the window used is large enough compared to the channel memory. The

estimate of one symbol at time $n$is then given by

$$\begin{aligned}\mathbf{a}_n &= Cov(\mathbf{r_n}, \mathbf{r_n})^{-1} Cov(\mathbf{r_n}, x_n) \\ b_n &= E\{x_n\} - \mathbf{a}_n^T E\{\mathbf{r}_n\},\end{aligned}$$

with the model in 2.2 we have

$$\begin{aligned}E\{\mathbf{r_n}\} &= \mathbf{H_n} E\{\mathbf{x_n}\} \\ Cov(x_n, \mathbf{r_n}) &= Var(x_n)\mathbf{h_n} \\ \mathbf{\Sigma}_n = Cov(\mathbf{r_n}, \mathbf{r_n}) &= \sigma_\omega^2 \mathbf{I} + \mathbf{H_n} Cov(\mathbf{x_n}, \mathbf{x_n}) \mathbf{H_n}^T,\end{aligned}$$

Where $\mathbf{h_n}$ is the $(N_2 + M)_{\text{th}}$ column of $\mathbf{H_n}$. Assuming the bits mapped to the symbols are independent, the symbols are then independent of each other hence the covariance matrix $\text{Cov}(\mathbf{x_n}, \mathbf{x_n})$ has zeros in all its off-diagonal elements and contains the variances of the symbols along the diagonal. With the *a priori* LLRs L_a^E the mean $\bar{x}_n$ and variance v_n of x_n are calculated as

$$\begin{aligned}\bar{x}_n &= \sum_{x \in S} x P(x_n = x) = P(x_n = 1) - P(x_n = -1) \\ &= \tanh(L_a^E(x_n)/2) \\ v_n &= E\{x_n^2\} - (E\{x_n\})^2 = 1 - \mid \bar{x}_n \mid^2,\end{aligned}$$

Defining $\mathbf{f_n} = \mathbf{\Sigma_n^{-1}} \mathbf{h_n}$, the estimate $\hat{x}_n$ can be expressed as

$$\hat{x}_n = \bar{x}_n + v_n \mathbf{f_n}^T (\mathbf{r_n} - \bar{\mathbf{r}}_\mathbf{n})$$

The extrinsic information from the equalizer should be independent of the *a priori* LLRs. This is achieved by setting $L_n = 0$ thus making $v_n = 1$ and $\bar{x}_n = 0$ in the computation of $\hat{x}_n$ This leads to modification of the filter coefficients $\mathbf{f_n}$ and the estimate $\hat{x}_n$ to

$$\begin{aligned}\mathbf{f'_n} &= (\mathbf{\Sigma}_n - (1 - v_n)\mathbf{h_n}\mathbf{h_n}^T)^{-1}\mathbf{h_n} \\ \hat{x}_n &= \mathbf{f'_n}^T(\mathbf{r_n} - \bar{\mathbf{r}}_\mathbf{n} + \bar{x}_n \mathbf{h_n})\end{aligned} \tag{3.1}$$

defining $s_n = \mathbf{f_n^T h_n}$, $\hat{x}_n$ can then be expressed as

$$\hat{x}_n = \frac{\mathbf{f_n^T}(\mathbf{r_n} - \mathbf{H_n}\bar{\mathbf{x}}_\mathbf{n}) + v_n s_n}{1 + (1 - v_n)s_n}$$

Here the matrix inversion lemma is used to simplify the expression for $\mathbf{f'_n}$. The extrinsic LLRs can be calculated using the statistics of the estimation error $e_n = \hat{x}_n - x_n$. The error is assumed to be Gaussian distributed with mean $E\{e_n\}$ and variance $Var(e)_n$ and

$$\begin{aligned}E\{e_n\} &= 0 \\ Var(e_n) &= (x_n) + Var(\hat{x}_n) - 2Cov(x_n, \hat{x}_n) \\ &= 1 - \mathbf{h_n^T}\left(\mathbf{\Sigma_n} - (1 - v_n)\mathbf{h_n}\mathbf{h_n^T}\right)^{-1}\mathbf{h_n} \\ &= 1 - \frac{sn}{1 + (1 - v_n)s_n}\end{aligned}$$

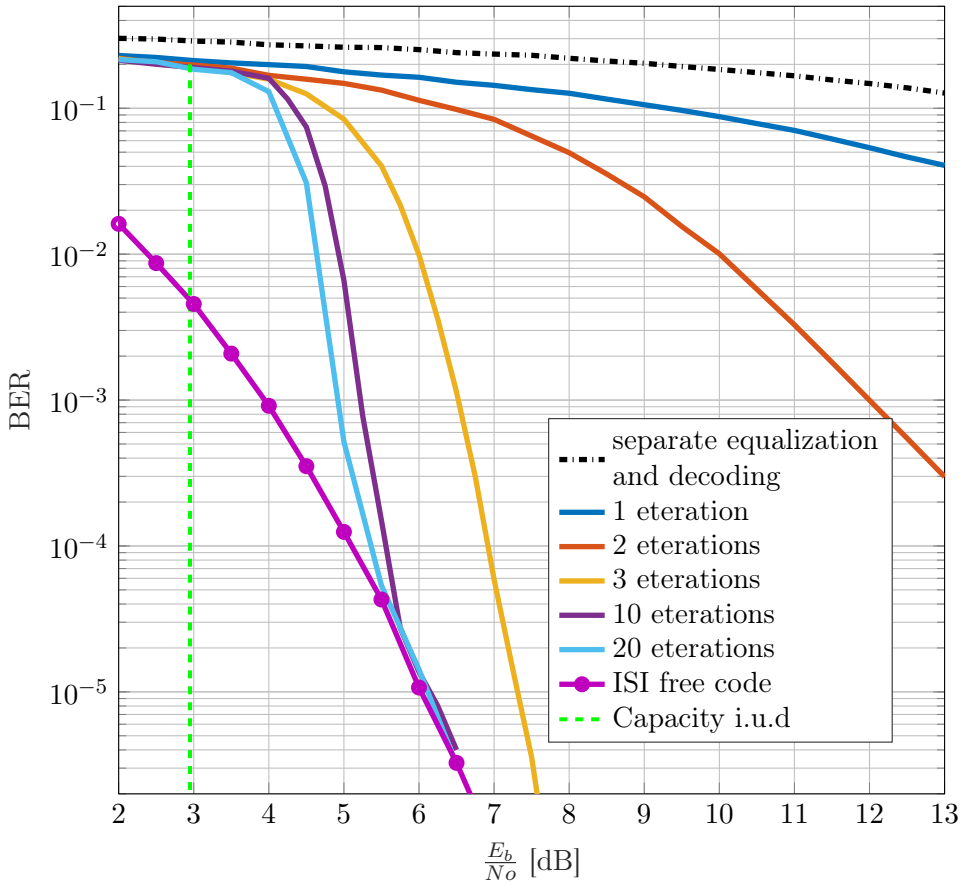

Figure 3.5: Results of turbo equalization with MMSE equalizer and convolutional code and interleaver length of 10000

The extrinsic LLR is then given by

$$\begin{aligned} L_e^E(x_n) &= \ln \frac{P(x_n = 1)}{P(x_n = -1)} = \ln \frac{P(e_n = \hat{x}_n - 1)}{P(e_n = \hat{x}_n + 1)} \\ &= \ln \frac{\exp\big(-(\hat{x}_n - 1)^2 / Var(e_n)\big)}{\exp\big(-(\hat{x}_n + 1)^2 / Var(e_n)\big)} \\ &= \frac{2\hat{x}_n}{1 - s_n/(1 + (1 - v_n)s_n)} \\ &= \frac{2(\mathbf{f_n^T}(\mathbf{r_n} - \mathbf{H_n}\bar{\mathbf{x}}_\mathbf{n}) + v_n s_n)}{1 - v_n s_n} \end{aligned} \tag{3.2}$$

In the computation of $\mathbf{f_n}$ we need to invert an $N \times N$ matrix for each n which has an order of complexity which is N^3. This can be reduced to $\mathrm{O}(N^2)$ by taking advantage of the structure of $\mathbf{\Sigma}_n$ which shows some time dependence [20]. Figure 3.6 shows the performance of turbo equalization with MMSE equalizer using the

$(1, 5/7)$ convolutional code as the component code. The performance of MMSE equalizer for the first few iterations is way below that of the MAP equalizer but with increasing iterations it also catches up with the MAP equalizer and eventually approaches the ISI free code.

The behaviour of the two types of equalizers can also be compared by examining their EXIT charts with the convolutional code. Despite not having a clear threshold but we observe some waterfall line behavior for which results into the performance approaching the limit imposed by the code. This occurs when the equalizer's EXIT curve has a significant opening at low mutual information points and the intersection only at points of relatively high mutual information. It can be seen that the MAP equalizer opens up its tunnel at 3.3 dB SNR while the LMMSE opens up at 4.3 dB.This behavior can be seen more clearly if a large interleaver is used as depicted in Figure 3.7 for an interleaver length of 10^5 This gives the MAP equalizer an advantage over the MMSE equalizer if the operating point is below 4 dB. But if thetarget BER rate is low such as 10^{-5} we can use the MMSE equalizer if the channel has longer channel memory thus reducing the complexity without sacrificing the performance.

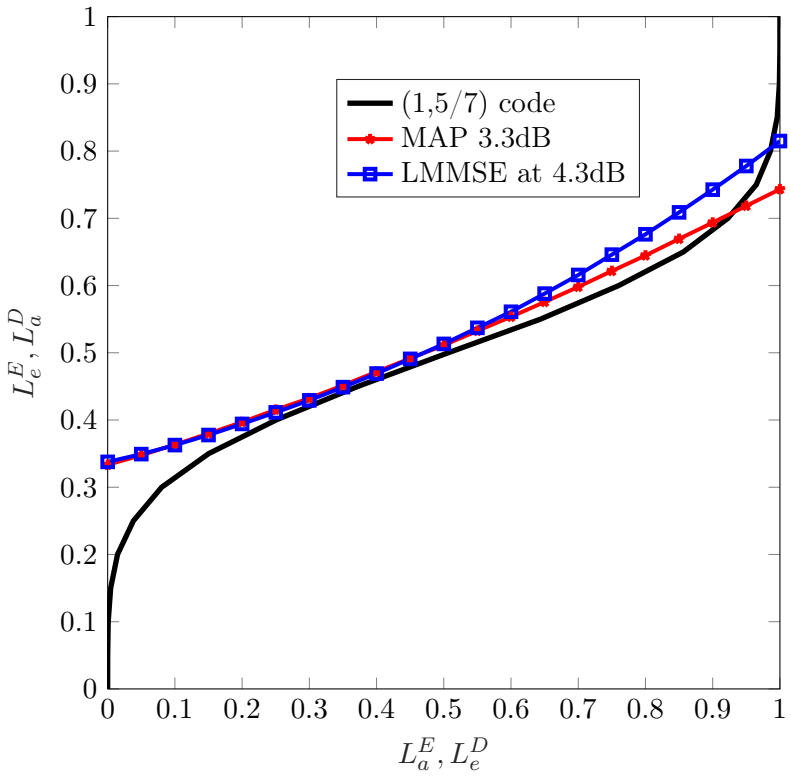

Figure 3.6: EXIT chart of MAP equalizer and LMMSE equalizer showing the point where the tunnel has significant opening

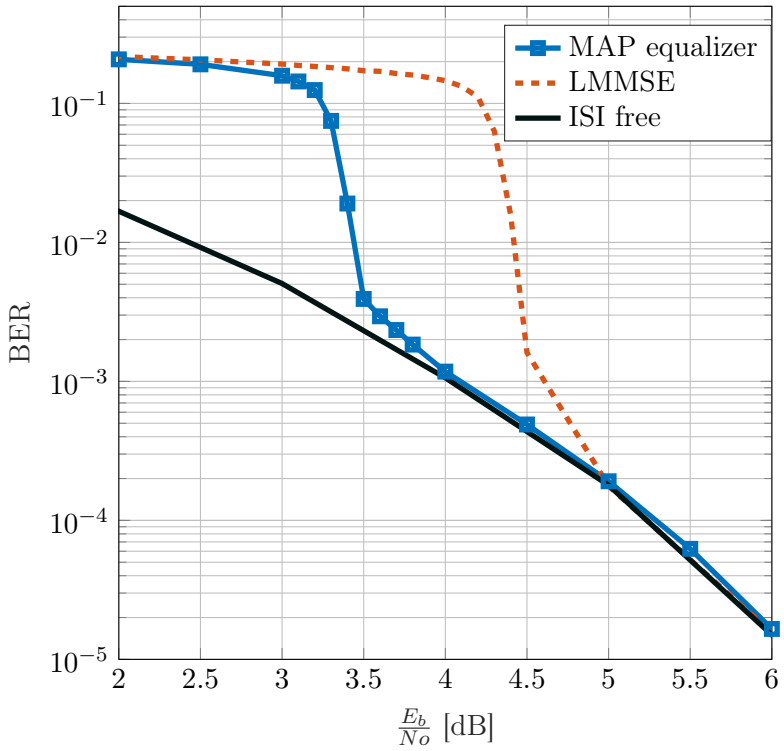

Figure 3.7: Comparison of the performance of MAP and LMMSE for interleaver length of 10^5 and 20 iterations demonstrating the effect of a significant open tunnel

3.2 Turbo equalization with LDPC code: Weak code versus strong code

In iterative systems there is always a trade-off between waterfall and error floor performance. This trade-off is also present in turbo equalization. This is illustrated by using a regular (3,6) and 5G LDPC code as the component codes with the same channel used for the (1,5/7) convolutional code. The factor graph for equalization with the MAP equalizer is shown in Figure 3.8 where the variable nodes are linked with the channel trellis through an interleaver. In the case of a the 5G code some nodes are punctured to obtain the desired rate. These nodes are shown in the graph as having no connection to the channel. In simulation, a turbo equalization system shows different behaviour if an all zero codeword is used instead of random data. This necessitates generating random bits and encoding them with an LDPC encoder. Generating sparse parity check matrix whose code can be encoded linearly is not a trivial task. To go around this we can use the fact that the channel performance does not depend on weather the input to it is a valid codeword of any code and the fact that for the code the distribution of the LLRs with respect to the input symbol(-1,+1) is what actually determines

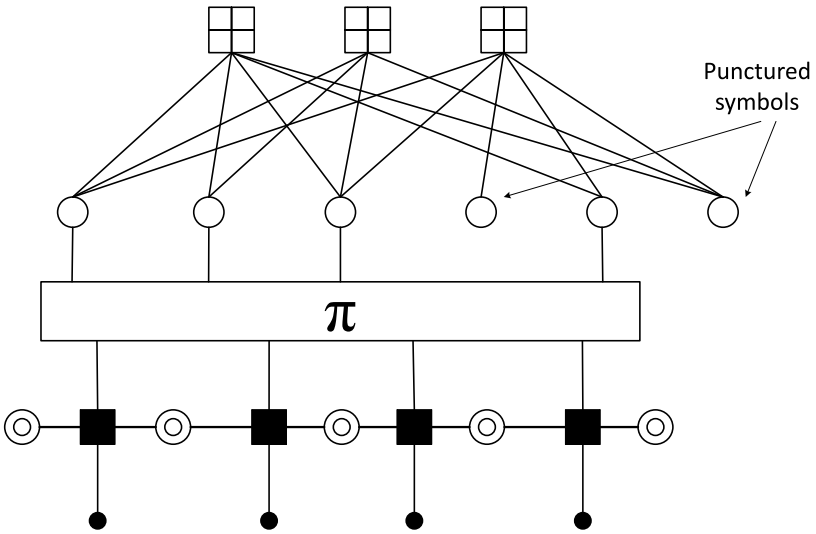

Figure 3.8: Factor graph of LDPC code in turbo equalization. Punctured symbols are shown with no connection to the channel graph

its response in an iterative system. That is for the code, the distribution of the LLRs from an all zero sequence is the same as the distribution of the variable $x \cdot L$ for any symbol x. The simulation is then done in the following manner. First

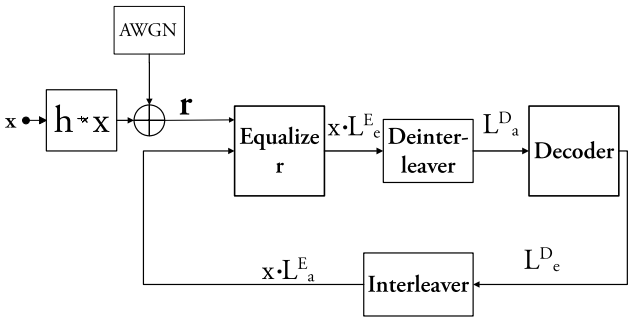

Figure 3.9: Simulation strategy of arbitrary code

a vector $\mathbf{x}$ of N random symbols is generated and feed as input to the channel (This is equivalent to generating an all zero sequence and scrambling it). Then the equalizer produces the extrinsic LLRs based on this sequence. The generated LLRs multiplied by $\mathbf{x}$ element-wise to and feed to the interleaver and then to the decoder. The decoder decodes uses these LLRs which now correspond to an all zero input sequence to produce its extrinsic LLRs. After interleaving the LLRs are then multiplied by their corresponding x to provide the aprior for the equalizer. This scheme is depicted in Figure 3.9. This was verified by simulation using some codes parity check matrix which could be encoded in a straightforward

manner like the 5G LDPC codes. The simulation result for turbo equalization

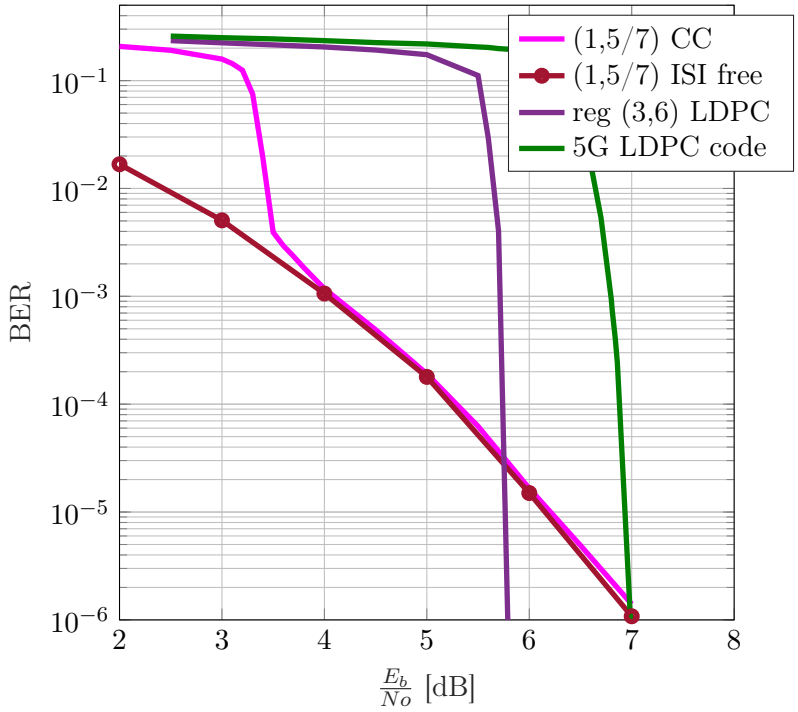

Figure 3.10: Comparison of turbo equalization with LDPC code and a convolutional code demonstrating the trade-off of choosing a weak versus a strong code

with regular (3,6) LDPC code with interleaver length of 43200 for 10 iterations is shown in Figure 3.10 alongside that of the convolutional code. We can see that the LDPC code has higher BER for SNR below 5.8 dB but it overtakes the ISI free convolutional code afterwards. This can be deduced from the EXIT chart of the LDPC code with channel in Figure 3.11. The LDPC code crosses the equalizer at a point of lower mutual information than the convolutional code for all SNRs below 5.5 dB. But once the SNR is reaches 5.5 the LDPC code does not cross the channel until it gets near perfect information. Since the tunnel is narrow at 5.5 dB the actual many iterations are required to cross it but since we have a limited interleaver length, the trajectory dies off before reaching the end due to correlation LLRs. After 5.8 the tunnel is wide enough for the trajectory to reach the perfect information points.

Figure 3.10 also shows simulation result with the 5G code obtained from base graph 2. With an interleaver length of 20480 obtained by using a lifting factor of 128. the interleaver length is obtained by concatenating 8 sections of 2560 bits each. This way of obtaining an interleaver shows better results than using a large uplifting factor for cases with limited interleaver length. This can be partially due to the fact that LLRs from the code are relatively more connected

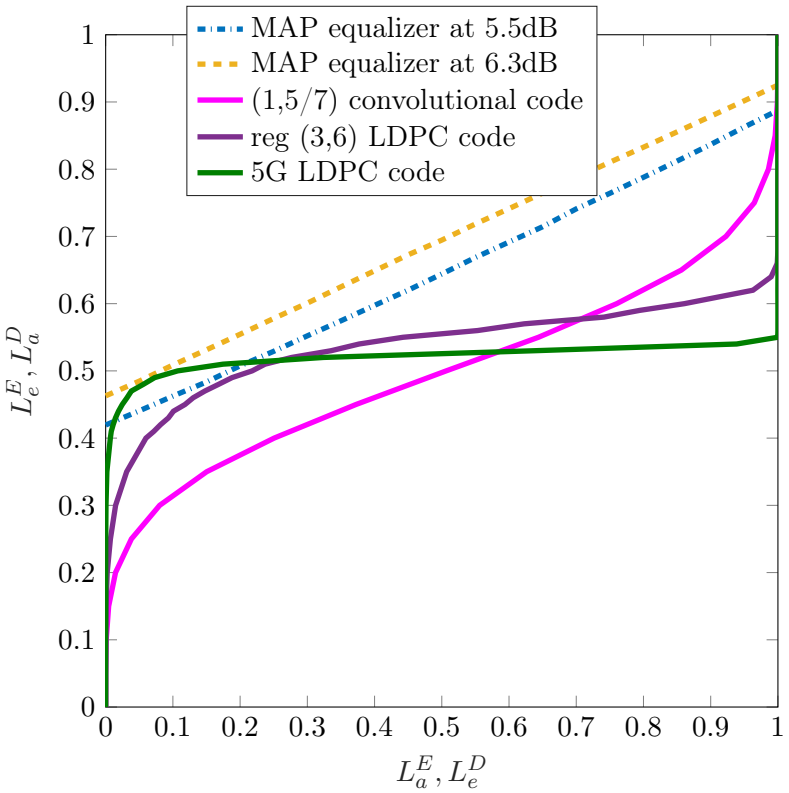

Figure 3.11: EXIT chart for (1,5/7) convolutional code, a regular (3,6) and 5G LDPC code with MAP equalizer

as the iterations in the code encounter cycles. Having an interleaver run through a concatenation of blocks will produce LLRs from unconnected sub-graphs thus reducing the correlations. Another reason is the change in the EXIT curve with uplifting factor which becomes insignificant as both lifting factors are large enough. The 5G code shows has a poorer performance in the waterfall region which can be deduced from the EXIT chart which has an opening at 6.3 dB as opposed to 5.5 dB for the regular (3,6) code. An interesting observation is the area below the two code EXIT curves are both equal to 0.5 which is the rate of the codes. This is in line with the prediction by the area theorem for EXIT charts as described in [4].

Chapter 4

Spatial Coupling in Turbo Equalization

Spatial coupling introduces memory between symbols generated at different times. Instead of encoding separate blocks at different times the blocks are interconnected to form a chain. With spatial coupling the threshold of a BP decoder can be reduced without decreasing the minimum distance of the code. This provides a new outlook on the design of iterative system where the trade off between waterfall and error floor can be avoided [15]. In this chapter spatial coupling of serially concatenated codes and LDPC code is introduced followed a discussing on how it can be applied to turbo equalization

4.1 Spatial Coupling between the inner and outer code for SCC codes

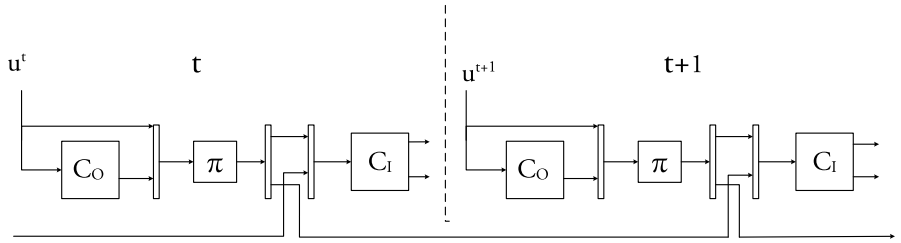

Figure 4.1: Spatially coupled SCC with $m = 1$

Spatial coupling of serial concatenated codes (SC-SCC) introduces memory between the outer code and the inner code such that the input to the inner encoder at time t depends not only on the output of the outer code at the time but also on parts m past blocks where m is the memory of the system. A case with memory $m = 1$ is used for illustration in Figure 4.1 following closely the work in [14] with minor modifications. At time $t$the outer encoder accepts information bits $\mathbf{u}_t$ and outputs $\tilde{\mathbf{v}}_t^o$ which is divided into two sub-blocks of equal length, $\tilde{\mathbf{v}}_{t,1}^o$ and $\tilde{\mathbf{v}}_{t,2}^o$. The inner code accepts two sub-blocks produced by the outer encoder at times t and $t-1$ namely $\tilde{\mathbf{v}}_{t,1}^o$ and $\tilde{\mathbf{v}}_{t-1,2}^o$. $\tilde{\mathbf{v}}_{t,2}^o$ is initialized to $\mathbf{0}$ for t<1. The sequence of blocks is terminated after L runs. The termination scheme used assumes $\mathbf{u}_t$ and

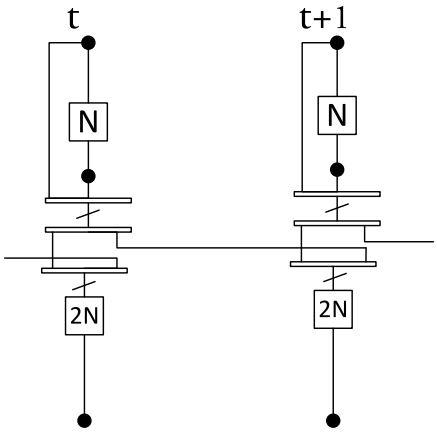

Figure 4.2: Compact graph representation for SC-SCC

the corresponding $\tilde{\mathbf{v}}_t^o$ to be zero for $t > L-m$. Since the outer encoder is recursive this is done by terminating the outer encoder to the zeros state at the end of section $L-m$ .This is achieved by inserting m_c termination bits depending on the final state, where m_c is the memory of the convolutional code. This introduces a rate loss as shown in (4.1) which becomes insignificant as L increases.

$$\begin{aligned} R &= \frac{N(L-m)-m_c}{4NL} \\ &= \frac{1}{4}\Big(1-\frac{m}{L}\Big)-\frac{m_c}{4NL} \end{aligned} \tag{4.1}$$

The term $\frac{m_c}{4NL}$ is negligible for large L.

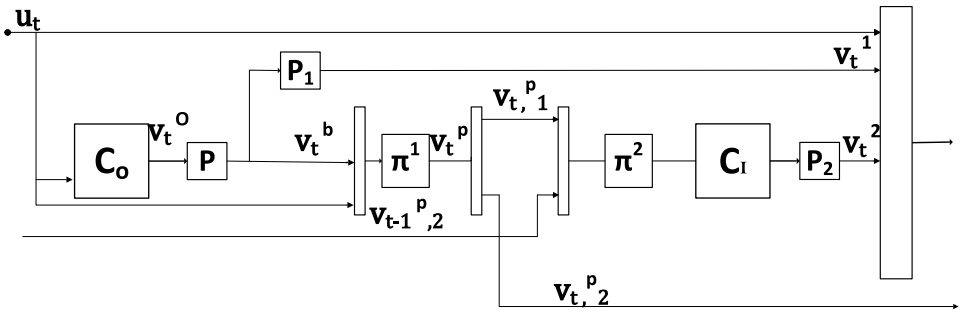

Figure 4.3: Coupling of a punctured SCC code

The code used in this thesis a rate 1/2 SCC code obtained by puncturing as described in Subsection 2.3.3. As illustrated in Figure 4.3 the outer code accepts $\mathbf{u}_t$ block of N data bits and produces N parity bits $\mathbf{v}_t^o$ which are punctured to obtain $N/2$ bits $\mathbf{v}_b^t$. $\mathbf{v}_b^t$ is multiplexed with $\mathbf{u}_t$ and interleaved to form $\mathbf{v}_t^p$. Coupling with memory 1 is achieved by splitting the $3/2N$ bits $\mathbf{v}_t^p$ into two sub-blocks of equal size $\mathbf{v}_{t,1}^p$. and $\mathbf{v}_{t,2}^p$. The inner code accepts $\mathbf{v}_{t,1}^p$ and $\mathbf{v}_{t-1,2}^p$ (from previous

time) after being multiplexed and interleaved by the interleaver Π^2. The $3/2N$ parity bits produced by the inner code are punctured by to obtain $N/2$ bits $\mathbf{v}_2^t$. Finally at time t, ($\mathbf{u}_t$,$\mathbf{v}_{tb}$,$\mathbf{v}_2^t$) are multiplexed and transmitted. At time $t+1$ the process is repeated with the encoders keeping track of their final states at time t. $\mathbf{v}_{t,2}^p$ is set to $\mathbf{0}$ for $t < 0$ and $\mathbf{u}_t$ is set to $\mathbf{0}$ for $t > L - m$. This provides known symbols which to the decoders at the receiver.

4.2 Coupling between interleaver and channel

Motivated by spatial coupling of serially concatenated codes and the fact that an ISI channel and the code form a serially concatenated system, we introduce memory between blocks from the interleaver and the mapper (effectively introducing memory between the interleaver and the channel). Spatial coupling in this case

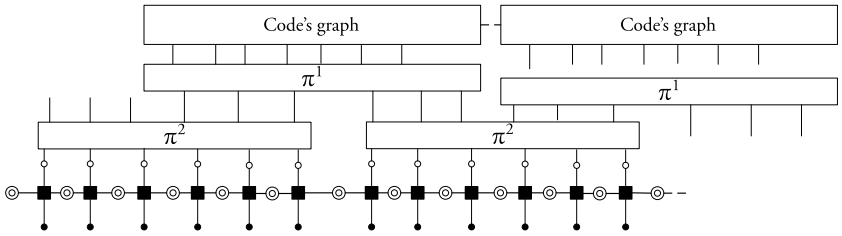

Figure 4.4: Factor graph representation of coupling between interleaver and channel

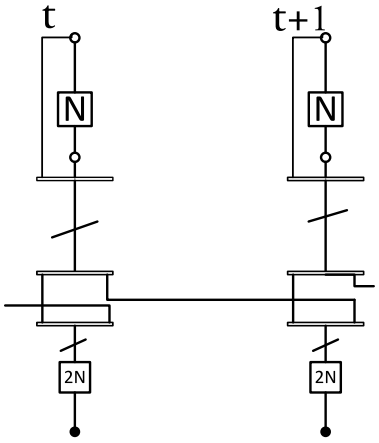

Figure 4.5: Compact graph representation of coupling between interleaver and channel for BPSK modulation

is done splitting of the encoder after the interleaver into $m+1$ sub-blocks. Figure **??** shows the factor graph representation of the scheme with $m = 1$ and BPSK modulation. The channel will always keep track of the states as we cannot restart

it as we wish hence the channel graph is connected for all blocks. For the code, on the other hand, the connection between blocks are in our control depending on the code type and design. Just as in the case of spatially coupled SCC, the output of the code at time t after interleaving is split into two blocks $\mathbf{v}_{c,1}^t$ and $\mathbf{v}_{c,2}^t$. The mapper at time t accepts $\mathbf{v}_{c,1}^t$ and $\mathbf{v}_{c,2}^{t-1}$ which are multiplexed and interleaved before being mapped into symbols $\mathbf{x}$ depending on the modulation scheme. The initialization and termination is done in the same way as described in the Section 4.1 by introducing known bits at the beginning and end of the chain of length L.

4.2.1 Window decoding of a spatially coupled SCC system

In this subsection window decoding of a coupled system with memory 1 is discussed with the channel as the inner component. As mentioned in the previous subsection the channel has perfect knowledge of half of the bits for the first block (theoretically corresponding to infinity LLRs). This enables the channel equalizer to make better decisions for the first block. This improved decision is picked up by the code resulting into even better beliefs. If this is repeated for some iterations the first block can be virtually error free. The same occurs for the last block. This wave of good LLRs travels along the chain as the number of iterations increase. Figure 4.7 illustrates the phenomenon for small coupled system with $L = 50$. The edges has low BER which falls and spreads out with iterations. For a usable system however this will be impractical as it would entail the need to wait for all the blocks to be received before making any decisions on any block thus making the latency unacceptable.

To effectively benefit from spatial coupling we need to use a window decoder. In this case only W blocks are needed to decode the first block in the window. In case of a MAP equalizer and an inner code decoded on a trellis this would imply bad β messages will be transmitted from the right end of the window (the states are initiated with equally likely distribution). But with a sufficiently big N and decoder size at least four blocks more than the memory this effect will be masked before reaching the first block. As mentioned before, the very first block will provide good LLRs which will help the code to correct most of the errors. this first block of the code is however linked to the second block of the channel and the second is linked with the third and so on. So when the window moves to the next block the channel will have the good LLRs from the previously first block which would then help in decoding the currently first block (which was the second block previously). In this fashion the whole chain can be decoded effectively to low BER with reasonable latency. To get good results it is good to have more iterations at the very first few blocks than the rest. This is for two reasons. The first being to ensure an almost error free decoding of these blocks and the second reason is that the actual number of iterations for blocks inside the chain is is W times the number of iterations per block as they will be covered during the decoding of W previous blocks. With this scheme, however, when erros occurs in one block the errors tends to propagate for some time in the chain. Figure 4.7 illustrates the idea window decoding of a channel with a code. The bits in the channel are depicted before being permuted by the second interleaver for illustration. In this thesis we use a window size of 5 for systems with $m = 1$ window size of 9 for $m = 2$.

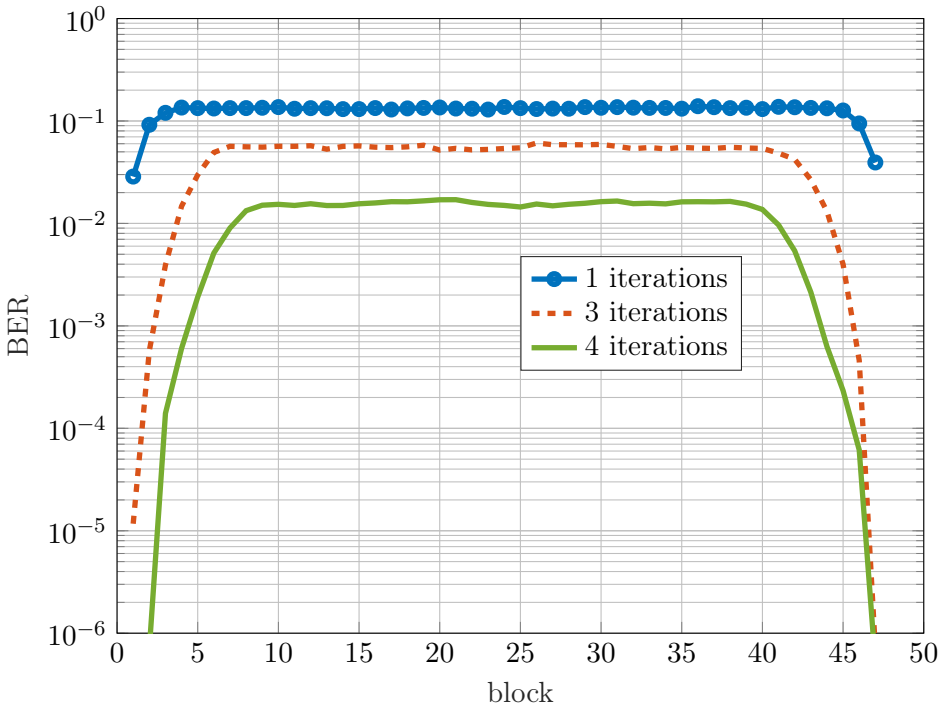

Figure 4.6: Decoding of a spatially coupled SCC system the effect of known bits with for coupling at channel interleaver with a regular (3,6) LDPC code as the component code

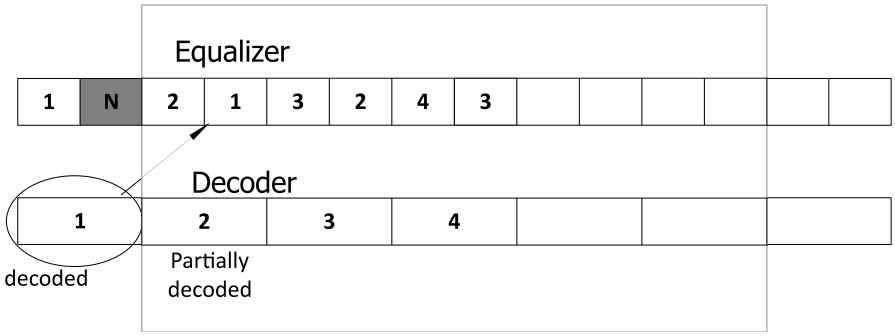

Figure 4.7: Window decoder for window size 5, decoding block 2. At this stage block 2 to 5 have some partial information obtained during decoding of block 1

4.3 Turbo equalization with spatial coupling

In this section different codes and coupling strategies are discussed with their simulation results.

4.3.1 Channel coupling with convolutional code

With the convolutional code as the component code coupling between the interleave and channel does not change the performance in any significant way. We only observe a slight gain at the low SNR but the error floor dictated by the code comes into play early. Figure 4.8 shows the simulation result of such a system.

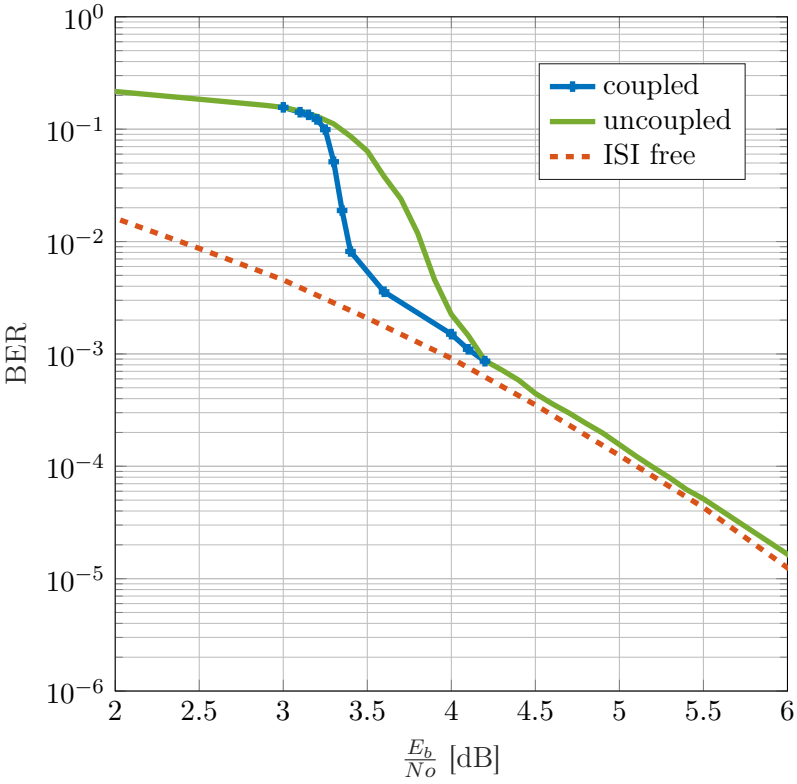

Figure 4.8: Simulation results for coupling between interleaver and channel using a convolutional code for interleaver length of 10000

4.3.2 Channel coupling with LDPC code

In this scheme an LDPC code is used as the component code in such a way that separately encode blocks are coupled after being interleaved. In this way there is no direct connection between the graphs of the codes in the chain but the connection is only at the channel graph. Using a regular $(3, 6)$ code we observe a gain of about 1.2 dB as depicted in Figure 4.9 which makes the LDPC code to overtake the convolutional code at about 4.5 dB by having BER as low as 10^{-5}. It is interesting to not that this happens at 1 dB below the threshold of the uncoupled turbo equalization system. The gain is relatively small but it provides way to

improve the performance of system using an LDPC code without worrying much on the code design which means we can apply this form of coupling to an existing system without making any significant changes.

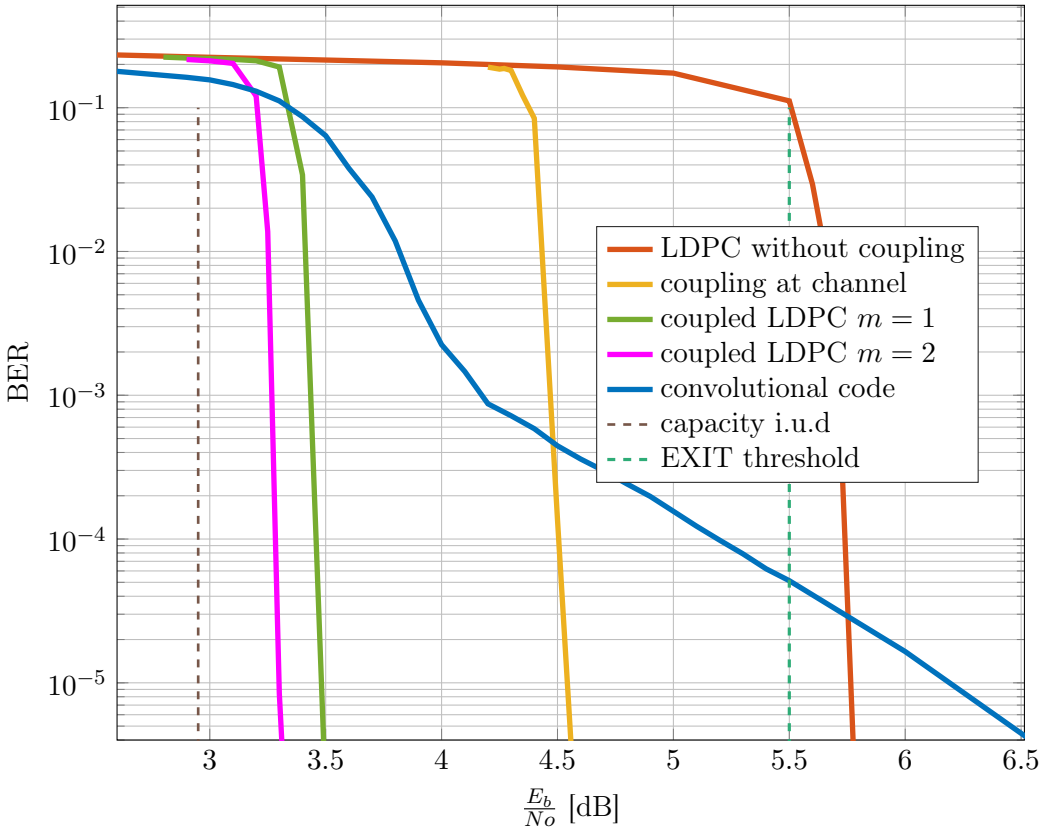

Figure 4.9: Results of coupling in turbo equalization with LDPC code illustrating the effect of coupling at channel and using a coupled code

4.3.3 SC-SCC code versus SCC coupling at channel

Coupling in turbo equalization with SCC code can be introduced in several ways. Two ways are compared with the first introducing memory between component codes as discussed in Subsection 2.3.3 and the second introducing memory between the channel interleaver and the channel as described in Subsection 4.2. The two schemes are depicted in compact graph representations in Figure 4.10. The simulation results are shown in Figure 4.11. It can be seen that both ways of coupling manages to obtain low BER rate below the EXIT threshold of the uncoupled system which is 5.5 dB. But more gain is observed in in the case with coupling at the interleaver with the channel. This might be the fact that the outer code is weak

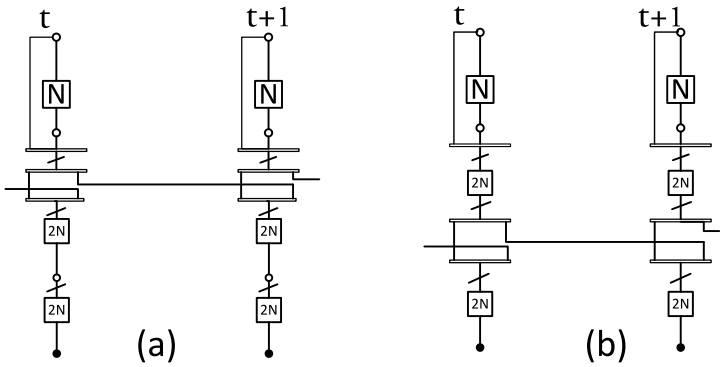

Figure 4.10: Compact graph representation of turbo equalization with SC-SCC (a) SCC coupling at channel (b)

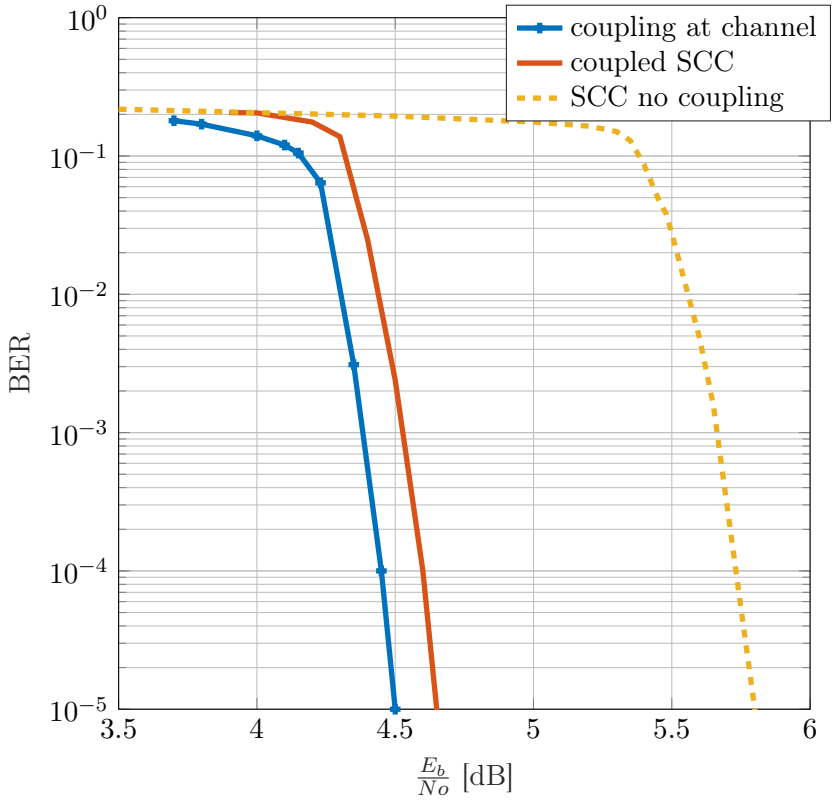

Figure 4.11: Simulation results of different ways of coupling a turbo equalization system with SCC code

and is further weakened by puncturing thus making it less effective in picking up the enhancement provided by the outer code while the SCC is a strong code thus

can effectively pick up the enhancement from the channel. Further investigation can is needed however to have a clear insight.

4.3.4 Spatial coupling of LDPC codes

For an LDPC code we can consider the check nodes as single parity check codes and the variable nodes as repetition codes. We can construct coupled LDPC codes from a protograph as follows. If variable node of type v_j is connected to check node of type c_i in the protograph, spatial coupling of memory m is done by spreading the edges from v_j over $m+1$ check nodes of type c_i from $t=0$ to $t=m$. This makes the single parity check code at time t depend not only on the symbols at time t but also on the previous m symbols. If we examine the base matrix we see that spatial coupling splits the matrix $\mathbf{B}$ into $m+1$ components $\mathbf{B}_0$ ... $\mathbf{B}_m$ such that $\mathbf{B}_0 + \mathbf{B}_1 \ldots \mathbf{B}_m = \mathbf{B}$. The chain can be terminated after length L. Figure 4.12 illustrates this with a regular (3,6) code with coupling memory $m=1$ and base matrices $\mathbf{B}_0$ and $\mathbf{B}_1$ given by

$$\mathrm{B}_0 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}, \qquad \mathrm{B}_1 = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

In general a coupled LDPC code with memory m and termination length L and

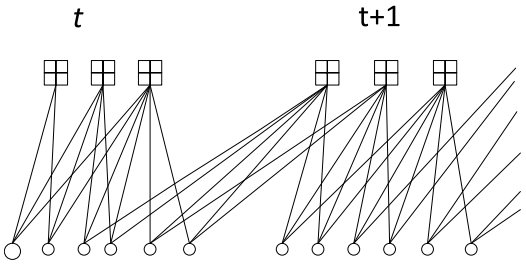

Figure 4.12: Protograph of a coupled regular (3,6) LDPC code

base matrices $\mathbf{B}_t$ of size $b_c \times b_v$ has the convolutional base matrix of the form

$$\mathbf{B}_{[0,L-1]} = \begin{bmatrix} \mathbf{B}_0 & & & \\ \mathbf{B}_1 & \mathbf{B}_0 & & \\ \vdots & \mathbf{B}_1 & \ddots & \\ \mathbf{B}_m & \vdots & \ddots & \mathbf{B}_0 \\ & \mathbf{B}_m & & \mathbf{B}_1 \\ & & \ddots & \vdots \\ & & & \mathbf{B}_m \end{bmatrix}_{(L+m)b_c \times Lb_v}, \tag{4.2}$$

Applying the coupled code to an ISI channel such that window decoding can be used results in a combined factor graph given by Figure 4.13. The interleaver

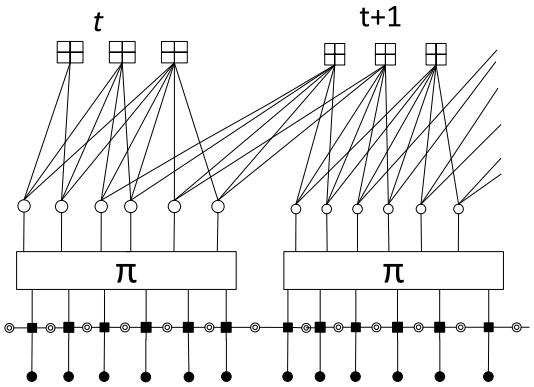

Figure 4.13: Graphical representation of turbo equalization with a coupled LDPC code

runs through symbols at time t before being mapped and put to the channel. To decode this system we use window decoding in the following manner. A window covering W blocks is accessed by the equalizer which produces its extrinsic LLRs. These extrinsic LLRs are fed to the LDPC decoder which updates its corresponding variable nodes and iterates within the codes graph between the variable and check nodes for a predefined number of cycles. The iterations can be stopped if the check equations connected to the variable nodes being decoded are satisfied to speed up the process. The iterations in the LDPC decoder needs access to m blocks of decoded symbols in addition to the W blocks it freshly received. It also need access to some symbols which has no information from the channel on the right end. Figure 4.14 illustrates window decoding with $m = 1$ and $W = 5$. The first blocks inside the window (dark) represent edges to nodes being decoded while faint blocks outside the window represent past edges involved in the current decoding. Simulation results show very good performance with gains gains more than 2 dB compared to the case without coupling as shown in Figure 4.9 where we observe BER below 10^{-5} for SNR around 3.55 dB 0.6 dB from capacity limit. Increasing the memory to 2 improves the performance further. One challenge with this method is for the encoding part, since we need to encode the whole coupled code in one go before transmitting which might have more complexity as the resulting parity check matrix is more likely not have a structure that facilitates efficient encoding.

4.4 Comparison of spatial coupling with irregular LDPC codes

To get good performance in the waterfall region for a turbo irregular codes are often employed to match the channel EXIT function. For example in [19] irregular LDPC codes are designed by optimizing the degree distribution such that the code can have good performance in the waterfall region as shown in Table 4.1 which can sometime be better than with spatial coupling in some channels by some fraction of dBs. This method shows success but has two drawbacks. The first one is the

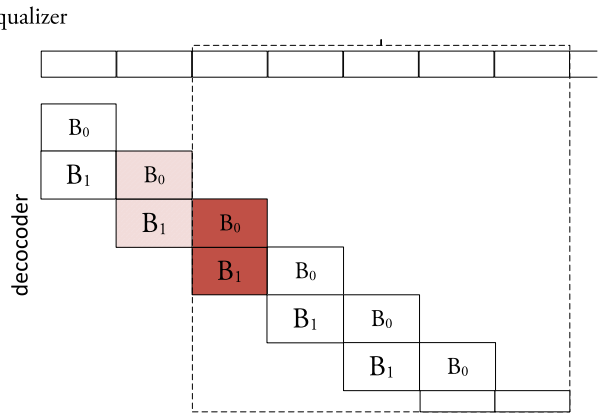

Figure 4.14: Illustration of window decoding of a coupled LDPC code for turbo equalization with a coupled LDPC code for channel 1

fact that it usually results into raised error floor. The second drawback is that it requires the knowledge of a particular channel which makes it unsuitable for channels which vary with time. Even if the channel does not vary much with time it is impractical for systems which are designed to work on any environment as it is difficult to predict the channel before deployment.

Further more we can still improve a spatially coupled system by introducing more memory as demonstrated by making the memory $m = 2$ in a system with coupled LDPC code.

channel	irregular codes	coupling $m = 1$	coupling $m = 2$
h_I	3.45 dB	3.5	3.3 dB
h_{III}	2.4 dB	2.6 dB	2.35 dB

Table 4.1: Comparison of spatial coupling with optimized irregular LDPC codes showing SNR at which the BER falls below 10^{-5}

Chapter 5

Conclusions and Future work

5.1 Conclusions

Turbo equalization has shown great success in mitigating the effect of ISI but as most iterative systems there is a trade-off between the error floor and waterfall performance.

This trade-off is demonstrated by comparing the performance of convolutional code and that of more strong codes like the SCC code and LDPC codes. The analysis from EXIT charts gave an accurate prediction of the limitations on choice of codes for turbo equalization. Since by the area theorem the area below the inverted codes transfer function is fixed to the rate of the code, changing a code while fixing the rate will almost inevitably result into a curve which either crosses the equalizer transfer function at point of low mutual information or at a point of high mutual information. in this thesis we investigated three ways to apply spatial coupling as a solution to this trade-off.

First spatial coupling is applied by introducing memory between the output of a code and the channel. Three codes were investigated for this type of coupling; a convolutional code, a serially concatenated code and a regular (3,6) LDPC code. With the convolutional code there was not much gain except a very little improvement in the performance in the low SNR region. The error floor limited by the performance of the code in an ISI free channel came into play quickly. With stronger codes, however, there was a noticeable gain of around 1.5 dB for both the SCC and LDPC code. Thus improving the performance of these codes in the waterfall significantly.

A second way by which spatial coupling was applied was by introducing memory between the outer and inner code for SCC code acting a componet code in turbo equalization. This also showed a significant gain but a little less than using a SCC with coupling at channel. A plausible reason for this could be the fact that the codes used for the SCC system were both weak convolutional codes which were punctured thus further weakening them.

Thirdly, a spatially coupled LDPC code was used as the component code. This form of coupling showed significant improvement in the performance in the waterfall region with gains more than 2 dB. The performance improved further by introducing more memory in the LDPC code with BER as low as 10^{-5} being

acheived only 0.3 dB from the capacity limit.

We showed on the other hand that to benefit from spatial coupling with reasonable latency we need to use window decoding. This is from the fact that to introduce spatial coupling and avoid the possible rate loss we need to use a long chain of blocks. To decode this chain without window decoding, the receiver would have to wait for all blocks to arrive which might introduce unacceptable latency.

5.2 Future work

Various aspects can be investigated further in regard to spatial coupling in turbo equalization.

Using SCC codes in turbo equalization results in an equivalent serial concatenation of three components; the outer code, the inner code and the ISI channel. By analyzing this concatenation in the framework of factor graph we can see a variety of ways of introducing coupling in the graph. Which coupling scheme is most effective? How does the choice of the component codes affect the performance? What schedule in updating the nodes in the resultant graph results into faster convergence? These and many other questions are still open to investigation.

Another aspect for further research is the performance of spatial coupling with higher order modulation. For example we can apply coupling between the code and channel by code bits after interleaving before being mapped to the higher order symbols. Some questions to be answered are: how does the bit mapping scheme such as grey coding or bit-interleaved coded modulation affect the performance of turbo equalization.

References

[1] Sílvio Almeida Abrantes. From BCJR to turbo decoding : Map algorithms made easier. 2005.

[2] Alexandre Graell i Amat, Fredrik Brännström, and Lars K. Rasmussen. On the design of rate-compatible serially concatenated convolutional codes. *European Transactions on Telecommunications*, 18(5):519–527.

[3] D. M. Arnold, H. . Loeliger, P. O. Vontobel, A. Kavcic, and W. Zeng. Simulation-based computation of information rates for channels with memory. *IEEE Transactions on Information Theory*, 52(8):3498–3508, Aug 2006.

[4] A. Ashikhmin, G. Kramer, and S. ten Brink. Extrinsic information transfer functions: model and erasure channel properties. *IEEE Transactions on Information Theory*, 50(11):2657–2673, Nov 2004.

[5] Z. Blazek, T. A. Gulliver, and V. K. Bhargava. On random interleavers for turbo codes. In *2001 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (IEEE Cat. No.01CH37233)*, volume 1, pages 111–114 vol.1, Aug 2001.

[6] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, New York, NY, USA, 2006.

[7] D. Divsalar and F. Pollara. Turbo codes for pcs applications. In *Proceedings IEEE International Conference on Communications ICC '95*, volume 1, pages 54–59 vol.1, June 1995.

[8] R. Gallager. Low-density parity-check codes. *IRE Transactions on Information Theory*, 8(1):21–28, January 1962.

[9] J. Hagenauer. The EXIT chart - introduction to extrinsic information transfer in iterative processing. In *2004 12th European Signal Processing Conference*, pages 1541–1548, Sep. 2004.

[10] S. Haykin and S.S. Haykin. *Adaptive Filter Theory*. Pearson, 2014.

[11] F. R. Kschischang, B. J. Frey, and H. . Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, Feb 2001.

[12] S. Lin and D.J. Costello. *Error Control Coding: Fundamentals and Applications*. Pearson-Prentice Hall, 2004.

[13] D. G. M. Mitchell, M. Lentmaier, and D. J. Costello. Spatially coupled LDPC codes constructed from protographs. *IEEE Transactions on Information Theory*, 61(9):4866–4889, Sep. 2015.

[14] S. Moloudi, M. Lentmaier, and A. Graell i Amat. Spatially coupled turbo-like codes. *IEEE Transactions on Information Theory*, 63(10):6199–6215, Oct 2017.

[15] Saeedeh Moloudi. *Spatially Coupled Turbo-Like Codes*. PhD thesis, Department of Electrical and Information Technology, 2018.

[16] J.G. Proakis. *Digital Communications*. McGraw-Hill, 1987.

[17] Tom Richardson and Rüdiger Urbanke. *Modern Coding Theory*. Cambridge University Press, 2008.

[18] S. ten Brink. Convergence behavior of iteratively decoded parallel concatenated codes. *IEEE Transactions on Communications*, 49(10):1727–1737, Oct 2001.

[19] Vladimir D. Trajkovic, Minyue Fu, and Peter Prof. Dr. Schreier. Turbo equalization with optimized linearly encodable ldpc codes. 2008.

[20] Michael Tüchler and Andrew C. Singer. Turbo equalization: An overview. *IEEE Trans. Information Theory*, 57(2):920–952, 2011.

Printed by Tryckeriet i E-huset, Lund 2019

LUND
UNIVERSITY

Series of Master's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2019-703
http://www.eit.lth.se