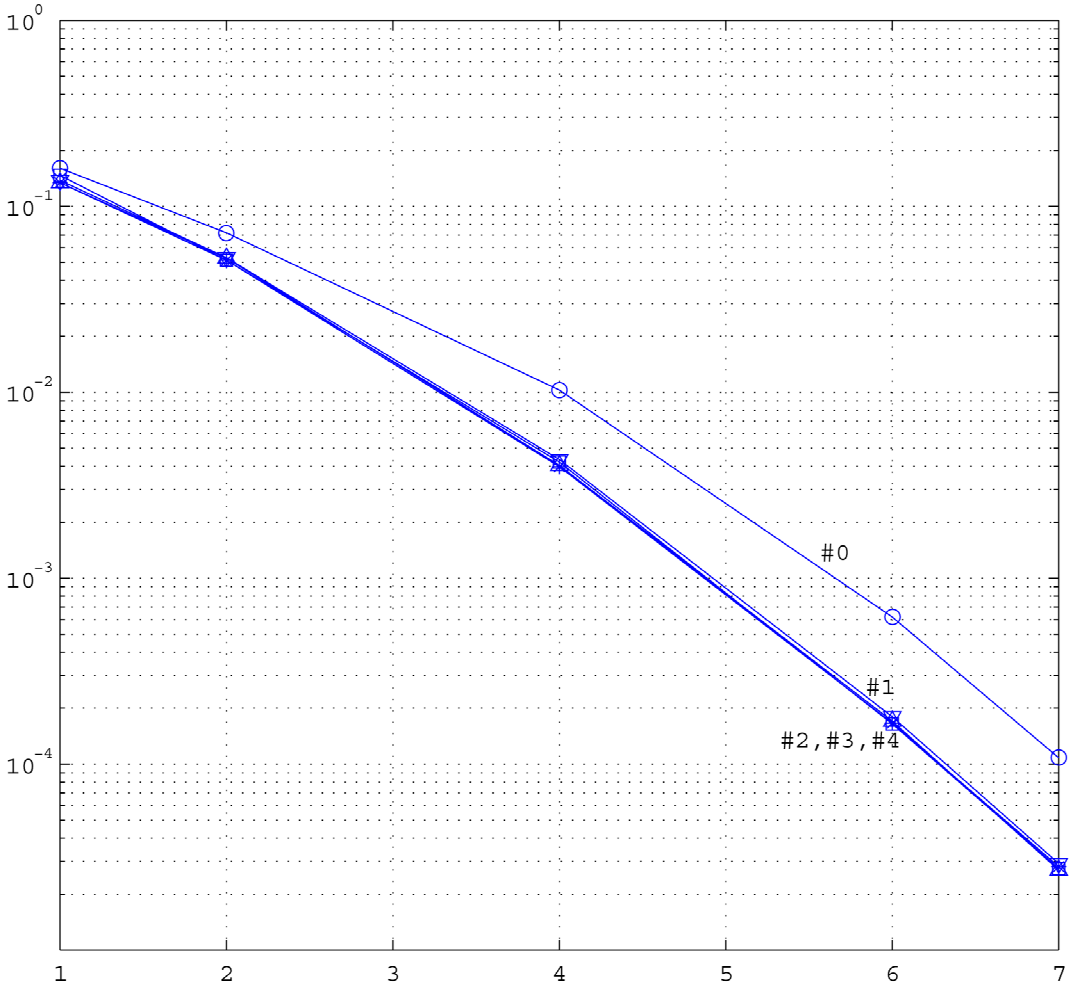
Improving GSM Receiver
by using
Iterative Equalization and Decoding

SIPCom - Group 892
8th semester

Spring 2005



AALBORG UNIVERSITY
The Faculty of Engineering and Science
Department of Communication Technology

Fredrik Bajers Vej 7, DK-9220 Aalborg Øst

Telephone +45 96 35 87 00

Aalborg University

The Faculty of Engineering and Science

Department of Communication Technology

Fredrik Bajers Vej 7 ▪ DK-9220 Aalborg Øst Telephone +45 96 35 87 00

Title:	Improving GSM Receiver by using Iterative Equalization and Decoding
Theme:	Methods and Algorithms
Project period:	February 1 - May 30, 2005
Specialization:	Signal and Information Processing in Communications

Project group:

892

Group members:

Kim Nørmark

Ole Lodahl Mikkelsen

Supervisor:

Bin Hu

Johan Brøndum

Romain Piton

Publications: 7

Pages:

Report	64
Appendix	19

Abstract

This report documents the design of an improved receiver for a GSM system. The receiver utilizes iterative equalization and decoding.
The system model consists of a modified GSM transmitter, a channel model, and a modified GSM receiver. The transmitter consists of a (23,33) convolutional encoder, a block interleaver, and a discrete time BPSK modulator. The channel is modelled as a multipath quasi-static block fading channel and a matched filter. The receiver consists of a Max-log-MAP equalizer, a deinterleaver, and a Max-log-MAP decoder implemented in an iterative manner.
Simulation and results show that there is a performance gain of 0.7 dB for BER at 10^{-3} by using iterative equalization and decoding for the first iteration. The following iterations do not give any appreciable performance gain. If fading is not produced due to multipath propagation in the channel, no irrecoverable bursts will occur and then there is a performance gain for each new iteration up till the fourth iteration. If the interleaver size is increased from 456 bits to 4104 bits, there is only little performance gain compared to size 456 bits at higher SNR and for the first iteration.

Preface

This report is conducted by group 892 of the *Signal and Information Processing in Communications* (SIPCom) specialization, 8th semester, Aalborg University. The project period spanned from february 1 to may 30, 2005, and the topic was "Improving GSM Receiver by using Iterative Equalization and Decoding" under the theme "Methods and Algorithms".

The group has in the project implemented a modified GSM transmitter, a channel model, and a modified GSM receiver. The receiver was improved by using an iterative technique.

This report is a documentation of the considerations made during this period. Furthermore, it documents the design of the considered system model and the simulations and results conducted.

The motivation for this project and for the considered system model is stated in the first chapter, the Introduction. Also the structure of this report is stated in the end of the Introduction.

An appendix are accompanying the report, concerning the implementation of the system model.

A CD is attached to the report. On the CD is the MatLab source code available, and also a ps file and a pdf file of the report is available.

Kim Nørmark	Ole Lodahl Mikkelsen

Contents

CHAPTER 1

Introduction

There has been a growing use of digital communication systems over the last five decades, and many applications today have a need for reliable and fast transmission through a channel. A few examples of applications could be systems for transmission of voice-, image-, or TV-signals, which we know and use in our daily life, and examples of channels could be wirelines for telephony or internet connections, fiber optic cables for music or video transmission, or the atmosphere for wireless transmission.

The basic idea with digital communication systems is to transmit a binary information sequence through a channel, and then receive it with as little error probability as possible. In order to do that, much signal processing is needed, both to transmit the information through the channel, and to protect the transmitted information to prevent errors in the estimated binary information sequence recovered from the received channel corrupted signal.

A digital communication system used for mobile telephony is the *Global System for Mobile Communications* (GSM), which is the most popular standard for mobile phones today. The GSM system transmits the digital information through the atmosphere by conveying the information to an analog waveform.

Since the GSM standard was introduced in 1982, there has been a major development in the field of digital communications design, and the requirements for the data transmission rate has increased. New methods for achieving better performance are needed, and the aim of this project is to improve the receiver part of a GSM system model. An often used process in GSM systems is a sequential equalization and decoding process, and a method to improve the performance of this process is to use an iterative technique, where the equalizer and the decoder are exchanging information with each other in an iterative manner.

To give a general understanding of how digital communication systems works, we will give a short introduction to digital communication systems in the first section of this chapter. The next section will give a short introduction to the GSM system with emphasize on the coding part, and in the third and last section of the chapter, we will discuss the implemented system model and state the assumptions and constraints we have made in the project.

The reference for this chapter is [Proakis et al., 2002] unless any other is specified.

1.1 Digital Communication Systems

The digital communication system consists of a transmitter and a receiver. In the transmitter, the binary information sequence in transformed into an electrical signal called the *message signal*, and the message signal is mapped into a carrier waveform suitable for transmission through the channel by a process called *modulation*. The modulated carrier waveform is transmitted through the channel and received in a degraded version by the receiver. In the receiver, the message signal is recovered from the received carrier waveform by a process called *demodulation*. Moreover, the receiver also performs amplification of the received signal, low pass filtering, *analog-to-digital conversion* (ADC), sampling, and noise suppression. In the front end of the receiver, some additive noise called *thermal noise* corrupts the transmitted signal in a random manner due to the amplification, which the system has to deal with.

The channel is the physical medium that is used to send the carrier waveform from the transmitter to the receiver. An essential feature of the channel is that it corrupts the transmitted signal due to man-made noise, interference from other users of the channel, and for wireless communications also from atmospheric noise picked up by the receiving antenna. In wireless channels, the phenomenon called *fading* may occur due to multipass propagation. It is a distortion of the transmitted signal and is characterized as a time variant amplification of the amplitude. The affect of all these corruptions and distortions must be taken into account in the design of the digital communication system.

So far we have described some of the transmitter, the channel, and the receiver. If we disregard the transmitting and receiving antennas, and the amplification, filtering, ADC, sampling, and noise suppression, the transmitter is basically just a modulator and the receiver is basically just a demodulator. The modulator maps a discrete (or digital) information sequence onto an analog carrier waveform, which is transmitted through the analog channel. The demodulator recovers the discrete information sequence by mapping the received analog carrier waveform into a discrete sequence. Digital communication systems often consists of more than a modulator and a demodulator, and a block diagram of the basic elements of a digital communication system is shown in figure 1.1.

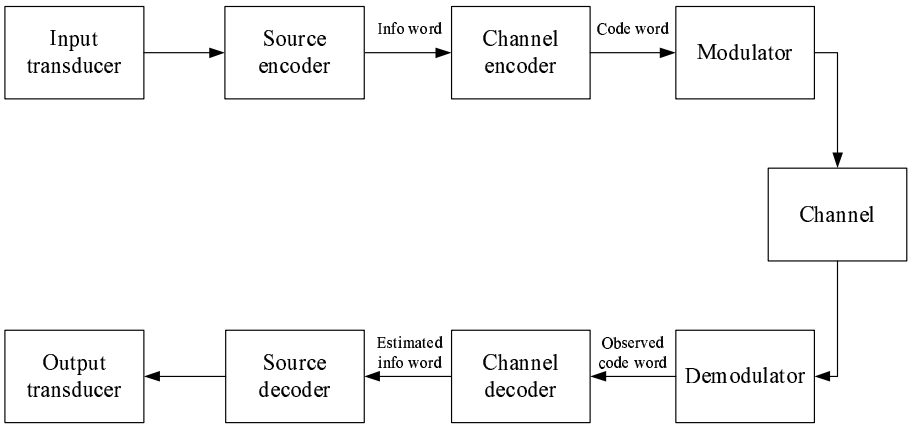


Figure 1.1: Basic elements of a digital communication system.

It is desirable to express the information with as few bits as possible in order to achieve fast data transmission. Therefore, instead of transmitting raw data from e.g. a speech signal or a video signal, a *source encoding* process is applied. In the receiver, a corresponding *source decoding* process is applied to re-create the raw data before the output transducer.

One of the most important properties of a digital communication system is to recover the original discrete information sequence with as few errors as possible. By using a *channel encoding* process, some redundancy bits are added to the discrete information sequence in a controlled manner. The input to the encoder is denoted as the *info word*, and the output is denoted as the *code word*. In the receiver, a corresponding *channel decoding* process is applied to re-create an estimated info word from the observed code word.

The channel coding gives an increased reliability of the received data and improves the fidelity of the received signal, but it also increases the number of transmitted bits per time instant and thereby the data transmission rate requirement. Also the modulation scheme chosen has effect on both the data transmission rate and the error probability, and the different modulation schemes are more or less suitable for different channels. These issues are a trade off the designer must have in mind when designing a digital communication system.

1.2 GSM Systems

The GSM standard for digital communication has been developed by ETSI[1] *Special Mobile Group* (SMG) since 1982. The group was first known as "Groupe Special Mobile", which was created by CEPT[2]. Groupe Special Mobile was transferred from CEPT to ETSI in 1992 and was renamed to ETSI SMG, and today the GSM standard is used by 254 million subscribers world-wide in over 142 countries around the world. [ETSI, 2004]

The GSM system is a digital cellular communication system that uses *Time-Division Multiple Access* (TDMA) to accommodate multiple users. European GSM systems uses the frequency bands 890-915 MHz and 935-960 MHz for uplink and downlink respectively, and each band is divided into 125 channels with a bandwidth of 200 kHz. Each of the 125 channels can accommodate 8 users by creating 8 TDMA non-overlapping time-slots of 576.92 μs, and 156.25 bits are transmitted in each time-slot with the structure denoted in figure 1.2. The data transmission rate is $\frac{156.25\,\text{bits}}{576.92\,\mu\text{s}}$ = 270.83 kbps.

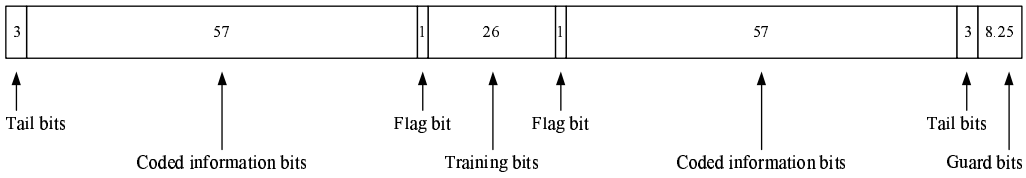


Figure 1.2: Time-slot structure for 1 user.

A block diagram of a GSM system is shown in figure 1.3. It consists of the same blocks as the block diagram for a general structure of a digital communication system shown in figure 1.1, but it also consists of more blocks to ensure proper and reliable transmission.

The *RPE-LPC speech coder* and the *Speech synthesizer* is the source coding/decoding part. The analog speech signal is in the RPE-LPC speech coder mapped into digital *Linear Predictive Coding* (LPC) coefficients by using the *Residual Pulse-Excited* (RPE) technique. In the Speech synthesizer, an estimation of the analog speech signal is constructed.

[1] European Telecommunications Standards Institute
[2] European Conference of Postal and Telecommunications

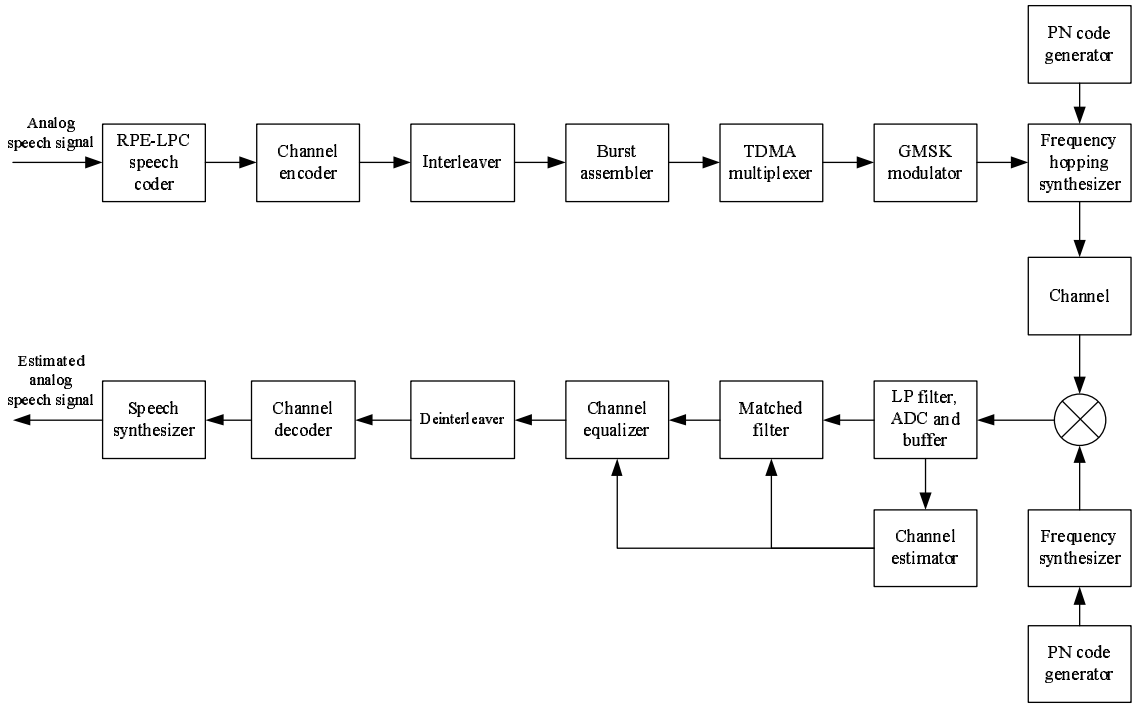


Figure 1.3: Block diagram of a GSM system.

The *Channel encoder* and the *Channel decoder* is the channel coding part. The channel encoder adds redundancy to the signal with linear convolutional block encoding, and the channel decoder decodes each block.

Linear convolutional block codes are designed to correct random errors, but in most physical channels the assumption that the noise appears randomly and independently is not valid. Often the errors tend to occur in bursts, and an effective method for correction of error bursts is to interleave the coded data. In this way the location of the errors appear randomly and is distributed over several code words rather than one. The *Interleaver* and the *Deinterleaver* is the interleaving part, which is not a part of figure 1.1.

The *Burst assembler* organizes the interleaved coded information bits for burst transmission carrying both the coded information bits and a "training sequence" to enable the receiver to estimate the characteristics of the channel.

The *TDMA multiplexer* distributes the bursts for the eight time-slots in the channel.

The *GMSK modulator* modulates the digital information onto an analog carrier waveform with *Gaussian Minimum-Shift Keying* (GMSK) technique.

In the *Frequency hopping synthesizer*, the analog carrier waveform is hopped to different frequencies controlled by the PN code generator, and the received signal is dehopped and translated to baseband by the *Frequency synthesizer*.

The *Lowpass filter, ADC, and buffer* is a part of the demodulator in figure 1.1. The signal is here converted to discrete time.

The *Channel estimator* uses the training sequence in the received bursts to estimate the characteristics of the channel for each time-slot and sends the coefficients to the matched filter and the channel equalizer.

The *Matched filter* seeks to maximize the SNR of the channel corrupted signal to suppress the noise.

The *Channel equalizer* uses *Maximum Likelihood* (ML) sequence detection to estimate the most likely interleaved code word.

1.2.1 The GSM Standard: Coding

The GSM standard defined in [3GPP, 2004] states that the input to the channel encoder is a data sequence, denoted as **d**, of 260 information (or info) bits. The elements in **d** are denoted as d_k and represent the LPC coefficients arranged after importance, where d_1 is most important and d_{260} is least important. The 182 first bits are protected by parity bits and are called $class$ 1, and the following 78 bits, called $class$ 2, are unprotected.

Reordering of Class 1 Information Bits

The class 1 information bits are reordered, and parity and tail bits are added. The new 189 bits long sequence, denoted as **o** where o_k is an element in **o**, is arranged as [3GPP, 2004]

$$o_k = d_{2k} \quad \text{for } k = 1, 2, \ldots, 91 \tag{1.1}$$
$$o_{184-k} = d_{2k-1} \quad \text{for } k = 1, 2, \ldots, 91 \tag{1.2}$$
$$o_k = p_k \quad \text{for } k = 92, 93, 94 \tag{1.3}$$
$$o_k = 0 \quad \text{for } k = 186, 187, 188, 189 \text{ (tail bits)} \tag{1.4}$$

where d_k are the class 1 information bits, and p_k are the parity bits.

The reordered class 1 information bits are shown as a graphical representation in figure 1.4.

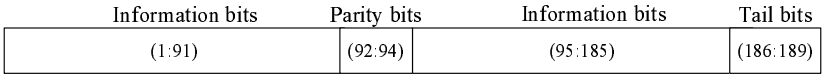


Figure 1.4: Graphical representation of the reordered class 1 information bits.

Encoder

The channel encoding scheme used in GSM systems is the (23,33) convolutional encoding scheme. Convolutional encoding is described in details in chapter 4.

The 189 reordered class 1 bits are encoded with a convolutional encoder with code rate R_c = 1/2 and defined by the polynomials:

$$g_1 = 1 + D^3 + D^4 \quad (1.5)$$

$$g_2 = 1 + D + D^3 + D^4 \quad (1.6)$$

which gives the (23,33) encoding scheme. 23 is the octal value of g_1, and 33 is the octal value of g_2.

The 189 class 1 bits give a sequence of 378 bits when encoded. The 78 uncoded class 2 bits are added to the 378 coded class 1 bits, which gives a code word sequence, denoted as **c**, of 378 + 78 bits = 456 bits. This is shown in figure 1.5 as a graphical representation. [3GPP, 2004]

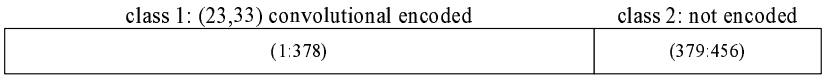


Figure 1.5: Graphical representation of the code word.

Decoder

The GSM standard does not specify any decoding algorithm, but an often used decoder in the GSM system is a ML decoder based on the Viterbi Algorithm. Convolutional decoding with the Viterbi Algorithm is described in details in chapter 5.

1.2.2 The GSM Standard: Interleaving

When the 456 coded bits are transmitted through the channel, they are block interleaved. The result of the interleaving is a distribution of the 456 code bits over 8 blocks of 114 bits, where the even numbered bits are in the first 4 blocks, and the odd numbered bits are in the last 4 blocks. Each interleaver block is transmitted as 2 half bursts, where each half burst consists of 57 bits. [3GPP, 2004]

The algorithm for the GSM interleaver is defined as [3GPP, 2004]

$$I_{B,j} = c_{m,k} \qquad \text{for } k = 1, 2, \ldots, 456 \quad (1.7)$$

where

$$m = 1, 2, \ldots \quad (1.8)$$

$$B = B_0 + 4m + ([k-1] \bmod 8) \quad (1.9)$$

$$j = 2[(49[k-1]) \bmod 57] + [([k-1] \bmod 8) \operatorname{div} 4] \quad (1.10)$$

where $I_{B,j}$ is a bit in a interleaver block, B is the burst number, and j is the bit number in burst B. m is the code word number, and k is the bit number in code word m.

One coded block is interleaved over 8 bursts starting from B_0. The next coded block is also interleaved over 8 bursts but starting from $B_0 + 4$.

1.2.3 The GSM Standard: Mapping in Bursts

Each burst consists of 2 half bursts of 57 bits and 2 flag bits indicating the status of even/odd numbered bits. The burst is transmitted in one time-slot in a channel for one user with the structure depicted in figure 1.2.

The mapping of the bursts is defined as:

$$e_{B,j} = I_{B,j} \quad \text{for } j = 1, 2, \ldots, 57 \tag{1.11}$$

$$e_{B,59+j} = I_{B,57+j} \quad \text{for } j = 1, 2, \ldots, 57 \tag{1.12}$$

$$e_{B,58} = hl_B \tag{1.13}$$

$$e_{B,59} = hu_B \tag{1.14}$$

where hl_B and hu_B are the flag bits for the B's burst.

1.3 The System Model

This section describes the system model considered in this project. We will go through some design considerations and focus on the assumptions and the constraints made in the project.

1.3.1 Channel Model Design

Mathematical Model

In the design of communication systems for transmission through a physical channel, it is important to have a good mathematical channel model that reflects the most important characteristics of the physical channel.

The simplest model is the *Additive White Gaussian Noise* (AWGN) channel, which is illustrated in figure 1.6. The transmitted signal, denoted as $x(t)$, is corrupted by an additive white Gaussian noise process, denoted as $n(t)$, which is the representation of the thermal noise from the components in the receiver. The output is described as $y(t) = x(t) + n(t)$.

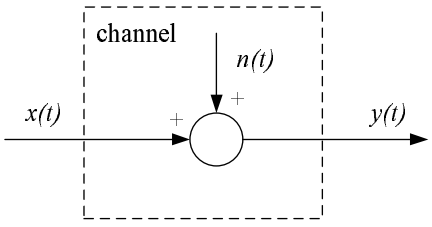


Figure 1.6: The additive white Gaussian noise channel.

Attenuation is often incorporated in the channel model, and the expression for the output is simply

$$y(t) = a \cdot x(t) + n(t) \tag{1.15}$$

where a is the attenuation factor. For $a = 1$ we have the AWGN channel, otherwise it is called a fading channel.

For some channels, it is important that the transmitted signal does not exceed a specified bandwidth limitation. Such a channel can be characterized as a time-invariant linear filter as depicted in figure 1.7. The channel output for such a channel is described as

$$\begin{aligned} y(t) &= x(t) * a(t) + n(t) && (1.16)\\ &= \int_{-\infty}^{\infty} a(\tau)x(t-\tau)\,d\tau + n(t) && (1.17) \end{aligned}$$

where $a(t)$ is the impulse response of the linear filter, and * denotes a convolution in time.

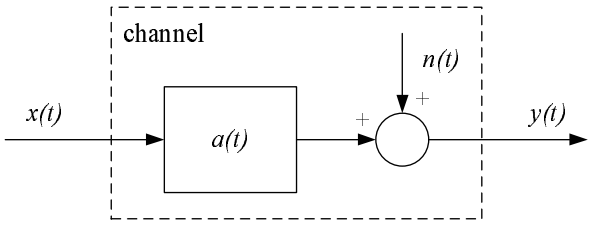


Figure 1.7: The linear filter channel with additive noise.

A third channel model is the linear time-variant filter channel, where the characteristics are time-variant due to time-variant multipath propagation of the transmitted signal. The impulse response of the filter is described as

$$a(t) = \sum_{i=1}^{L} a_i(t)\delta(t-\tau_i) \tag{1.18}$$

where the $a_i(t)$'s represent the time-variant attenuation factors for the L multipath propagation paths, and the τ_i's are the time-delays for the $a_i(t)$'s.

The channel output of the linear time-variant filter channel is described as

$$y(t) = \sum_{i=1}^{L} a_i(t)x(t-\tau_i) + n(t) \tag{1.19}$$

The affect of multipath propagation is a phenomenon called *InterSymbol Interference* (ISI). This issue is described in chapter 2.

The Sampling Process

In a physical channel, the digital data is transmitted by an analog carrier waveform, and the received waveform is demodulated to recover the analog message signal. The message signal is then

low pass filtered to remove irrelevant frequencies, sampled to represent it in discrete time, and quantized by an ADC to represent it as discrete values.

The sampling process converts the received analog signal into a corresponding sequence of samples that are uniformly spaced in time. If we consider an analog signal, denoted as $y(t)$, of finite energy and specified at all time, the sampling process of the signal can be described as [Haykin, 2001]

$$y_\delta(t) = \sum_{k=-\infty}^{\infty} y(kT_s)\,\delta(t - kT_s) \tag{1.20}$$

where T_s is the sampling time, $y_\delta(t)$ is the sampled signal of $y(t)$, and $\delta(t)$ is the delta function where $\delta(t) = 1$ for $t = 0$ and $\delta(t) = 0$ otherwise.

By expressing equation 1.20 in the frequency domain, the Fourier transform of $y_\delta(t)$ can be expressed as

$$Y_\delta(f) = \sum_{k=-\infty}^{\infty} y(kT_s)\,e^{-j2\pi kfT_s} \tag{1.21}$$

and by assuming that $y(t)$ is bandlimited and has no frequency components for $f > W$ Hz, where W is called the bandwidth of the signal, $Y_\delta(f)$ may by choosing $T_s = \frac{1}{2W}$ be rewritten as

$$Y_\delta(f) = \sum_{k=-\infty}^{\infty} y\left(\frac{k}{2W}\right) e^{\frac{-j\pi kf}{W}} \tag{1.22}$$

The spectrum of $Y_\delta(f)$ is shown in figure 1.8, where $f_s = \frac{1}{T_s}$ is the sampling rate.

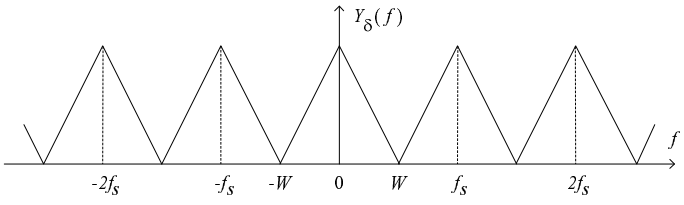


Figure 1.8: Spectrum of $Y_\delta(f)$ for $T_s = \frac{1}{2W}$.

By inspecting figure 1.8 it is clear that if $f_s < 2W$, the triangles will overlap. This will produce the unwanted phenomenon called *aliasing*. To avoid aliasing, the sampling rate must be greater than or equal to the so-called *Nyquist rate* ($f_s \geq 2W$). [Haykin, 2001]

Aliasing is a phenomenon considered in the frequency domain. The corresponding phenomenon in the time domain is ISI.

Channel Capacity

The bandwidth of the channel defines the limit for reliable data transmission through a channel, and the term *reliable transmission*, defined by Shannon in 1948, describes when it is theoretical possible to achieve error free transmission by appropriate coding. The channel capacity can be

expressed as

$$C = W \log_2 \left(1 + \frac{P_s}{W N_0}\right) \qquad [\text{ bits/s }] \tag{1.23}$$

where W is the channel bandwidth, P_s is the signal power constraint, and N_0 is the power spectral density of additive noise associated with the channel. Reliable transmission is achievable when the data transmission rate is less than the channel capacity.

By increasing the channel bandwidth, the channel capacity will increase also, but there is a limit for the capacity for a given channel, and this limit is called the *Shannon limit*. It is desired to get as close to the Shannon limit as possible to achieve the maximum performance for a given channel.

1.3.2 Assumptions and Constraints

The aim (or main scope) of this project is to improve the receiver in a GSM system model by using iterative equalization and decoding, but due to the time constraint in the project and to the fact that not all blocks are of equal relevancy, we will not implement a complete GSM system model.

The following elements has been disregarded compared to the GSM system depicted in figure 1.3:

Source Coding: Speech coding has only little relevance to the main scope, thus the RPE-LPC speech coder and the Speech synthesizer will not be implemented.

Multiuser System: Since the multiuser aspect is not in the scope of this project, the TDMA multiplexer and the frequency hopping will not be implemented.

GMSK Modulation: The modulation process is used to transmit the digital information through the channel and can not be omitted, but due to the time constraint, we will consider a simpler modulation process, namely the *Binary Phase Shift Keying* (BPSK) modulation.

Continuous Time Bandpass Transmission: A BPSK modulated signal is an analog carrier waveform where the digital information is conveyed by the phase. We consider a digital communication system with emphasis on the coding part and not on the modulation part, and we will therefore only consider a representation of BPSK modulation in a digital environment.

Channel Estimation: Due to the time constraint, the channel estimator will not be implemented. In stead we assume to have perfect knowledge of the channel at all time.

This project will consider the blocks depicted in figure 1.9, where we assume that the input to the system model is a discrete information sequence generated by a random generator. This sequence is also called the info word and is denoted as **d**. The following channel encoder is implemented as a (23,33) convolutional encoder, and where the GSM standard defines 182 class 1 bits and 78 class 2 bits in the info word, and only encodes the class 1 bits + parity and tail bits, the implemented encoder in the system model encodes a info word of 224 bits + 4 tail bits. The output sequence of 456 bits is called the code word and is denoted as **c**.

The code word is input to the *Interleaver + burst assembler*, where the code word is block interleaved with a modified GSM interleaver and mapped into four bursts per transmission. In stead of interleaving one coded block of 456 bits over 8 bursts in 16 half bursts of 57 bits as the GSM

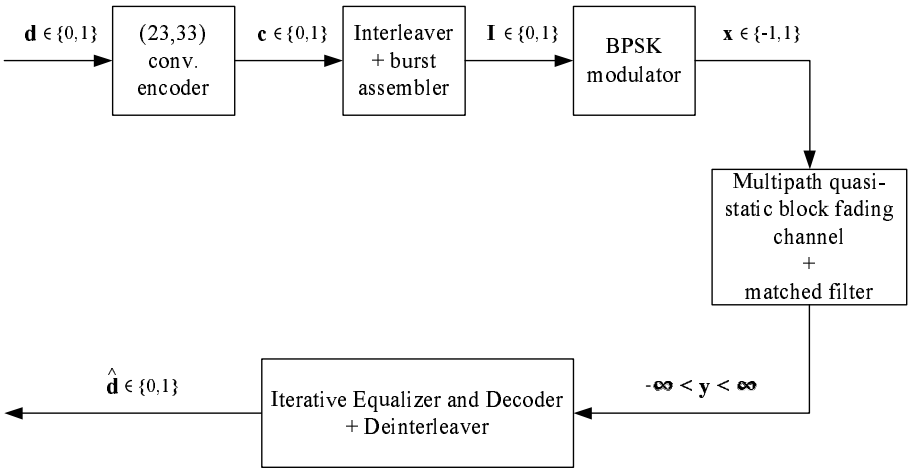


Figure 1.9: Block diagram of the system model.

interleaver does, the modified GSM interleaver interleaves one code block over only 4 bursts in 4 full bursts of 114 bits. Furthermore, the 4 bursts are transmitted sequentially and does not contain the two flag bits, hl and hu. The interleaved coded bits sequence is denoted as $\mathbf{I} = [\mathbf{e_1}\ \mathbf{e_2}\ \mathbf{e_3}\ \mathbf{e_4}]$, where $\mathbf{e_1}$, $\mathbf{e_2}$, $\mathbf{e_3}$ and $\mathbf{e_4}$ are the four burst sequences.

We consider discrete time BPSK modulation represented as a baseband signal, where the discrete input elements in $\{0, 1\}$ are mapped into one of the two output symbols -1 and 1. The input to the BPSK modulator is the four burst sequences of 114 bits and is denoted as **I**. The mapping is defined as

$$x_{B,i} = \begin{cases} -1 & \text{for } e_{B,i} = 0 \\ +1 & \text{for } e_{B,i} = 1 \end{cases} \tag{1.24}$$

where $x_{B,i}$ and $e_{B,i}$ is the i'th element in the B'th burst in **x** and **I** respectively.

We consider four channels, one for each of the four bursts transmitted. Each channel consists of a linear filter and corruption by AWGN as illustrated in figure 1.7. The linear filter has time-invariant attenuation factors due to multipath propagation, thus the name *quasi-static block fading channel*. The time-delays between the paths are fixed to one symbol duration. Due to the multipath propagation paths, ISI is produced in the channel. Finally, the channel model also consists of the matched filter.

We assume that ISI is not produced in the sampling process at the receiver. Furthermore, we assume that the symbol duration is equal to the sampling time ($T_{symbol} = T_{sample}$).

The GSM standard does not specify any channel decoding algorithm, but a commonly used decoding scheme is ML decoding using the Viterbi Algorithm. We seek to improve the performance in the receiver of a GSM system, and a technique that could improve the performance considerably, is *Iterative Equalization and Decoding*. A receiver using this technique is implemented and is described in details in chapter 7. The output of the receiver is hard information on the transmitted info word.

1.3.3 Structure of the Report

We have presented the implemented system model, which is considered in this project, and in chapter 2, the considered channel model will we described in further details. After the channel is described, the focus will be on the modulation process in chapter 3 with emphasis on the BPSK modulation scheme. Chapter 4 and 5 will describe the (23,33) convolutional encoding scheme and the corresponding decoding scheme using the Viterbi Algorithm. Since we have a multipath channel which produces ISI, equalization is relevant and is described in chapter 6. Chapter 7 will introduce the iterative equalization and decoding process and performance results will be presented. In chapter 8, we will present a conclusion and a discussion of the iterative equalization and decoding technique applied in a GSM system model.

CHAPTER 2

Channel Model

The channel in communication systems is the physical medium that is used to transmit a signal from the transmitter to the receiver. For the GSM system, where wireless transmission is used, the channel is the atmosphere, and it has the very important feature that it corrupts the transmitted signal. The most common corruption is thermal noise, which is generated in the receiver, and it comes in form of additive noise. Another corruption in wireless communications comes from signal attenuation (amplitude and phase distortion), and yet another comes from multipath propagation, which often results in *InterSymbol Interference* (ISI). ISI tends to occur in digital communication systems, when the channel bandwidth is smaller than the transmitted signal bandwidth, and it occurs when the transmitted signal arrives at the receiver via multiple propagation paths at different delays.

The channel model considered is a multipath quasi-static block fading channel with corruption by *Additive White Gaussian Noise* (AWGN). In order to minimize the affect of the channel corruption, a matched filter is applied after the channel.

The input to the channel is the BPSK modulated burst sequences of length 114 symbols, denoted as $\mathbf{x}$ in $\{-1, +1\}$. The output of the matched filter is sequences of 114 soft-valued symbols, denoted as $-\infty < \mathbf{y} < \infty$.

This chapter decribres first the AWGN channel and then the multipath fading channel. Finally there will be a describtion the matched filter. The primary reference for this chapter is [Proakis et al., 2002].

2.1 The AWGN Channel

In the AWGN channel, we consider the channel model shown in figure 2.1 The output is expressed as

$$r(t) = x(t) + n(t) \tag{2.1}$$

where $x(t)$ is the input signal and $n(t)$ is the white Gaussian noise process.

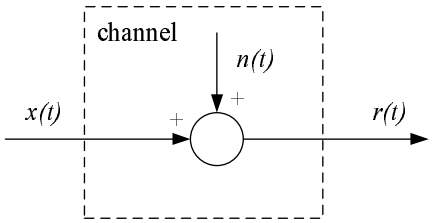


Figure 2.1: The AWGN channel.

2.1.1 Properties of AWGN

The noise is described in terms of its two parameters μ_n and σ_n^2

$$\mathcal{N}(\mu_n, \sigma_n^2) = \mathcal{N}(0, \frac{N_0}{2}) \tag{2.2}$$

where μ_n is the mean, σ_n^2 is the variance, and $\frac{N_0}{2}$ is the noise power spectral density.

The power spectral density function of the noise, denoted as $S_n(f)$, is illustrated in figure 2.2.

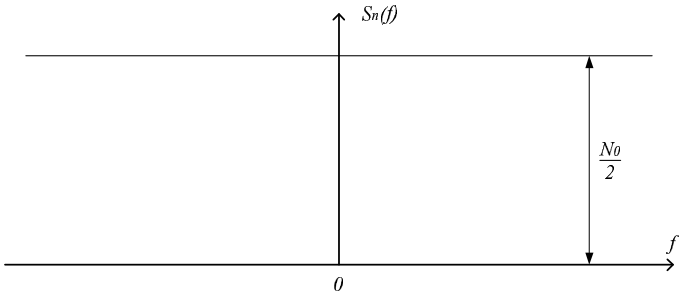


Figure 2.2: The power spectral density function of white Gaussian noise.

The white Gaussian noise is Gaussian distributed with the probability density function, denoted as $f_n(n)$, and can be illustrated as in figure 2.3.

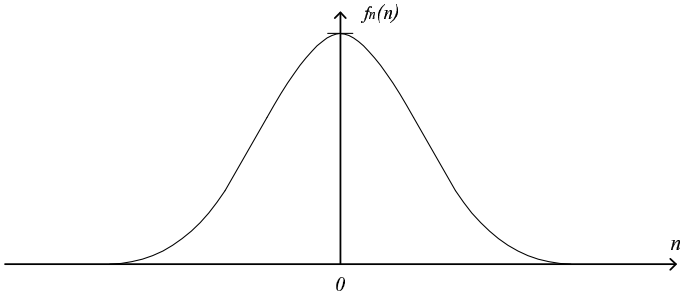


Figure 2.3: The probability density function of the white Gaussian noise.

2.1.2 Error Probability

When transmitting through the AWGN channel, the *Bit Error Probability* (BER) decreases for increasing SNR. This is shown in the BER plot in figure 2.4, which shows simulation results compared to the theoretical lower bound for transmission through the AWGN channel with BPSK modulation scheme.

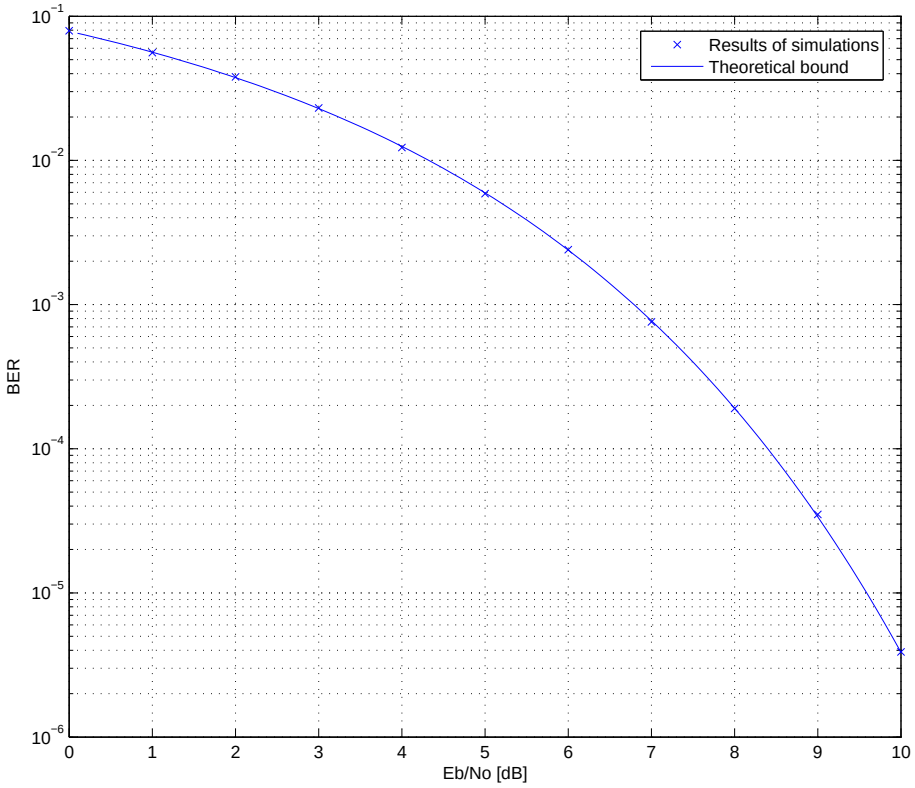


Figure 2.4: BER plot of the AWGN channel with BPSK.

The lower bound is the average bit-error probability

$$P_b = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) \tag{2.3}$$

where $\frac{E_b}{N_0}$ denotes the SNR. The derivation of equation 2.3 is shown in chapter 3 on page 22.

2.2 The Multipath Fading Channel

Wireless communication channels have often time-varying transmission characteristics and can be characterized as time-variant linear filters. This means that the channel impulse response is time-varying. Furthermore, the transmitted signal is usually reflected from the surroundings such

as buildings, which gives multiple propagation paths from the transmitter to the receiver with different time-delays and different attenuations.

The implemented channel model is a multipath quasi-static block fading channel implemented as a tapped delay-line, where the tap coefficients are modelled as complex-valued, Gaussian random processes which are mutually uncorrelated and time-invariant for each transmitted burst.

2.2.1 Single Path Fading Channel

We consider a single path fading channel as depicted in figure 2.5, where the attenuation factor (or tap coefficient), denoted as $a(t)$, is characterized as a complex-valued Gaussian random process defined as

$$a(t) = a_{re}(t) + ja_{im}(t) \tag{2.4}$$

where $a_{re}(t)$ and $a_{im}(t)$ are stationary and statistically independent real-valued Gaussian random processes.

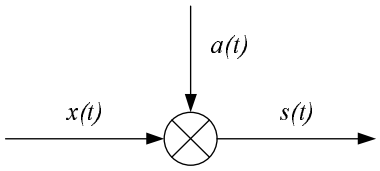


Figure 2.5: Block diagram of a single path fading channel.

$a(t)$ from equation 2.4 can also be expressed as

$$a(t) = \alpha(t)e^{j\phi(t)} \tag{2.5}$$

where $\alpha(t)$ and $\phi(t)$ are defined as

$$\alpha(t) = \sqrt{a_{re}^2(t) + a_{im}^2(t)} \tag{2.6}$$

$$\phi(t) = \arctan\frac{a_{im}(t)}{a_{re}(t)} \tag{2.7}$$

If $a_{re}(t)$ and $a_{im}(t)$ are Gaussian with zero-mean values, then $\alpha(t)$ is a Rayleigh distributed amplitude, and $\phi(t)$ is a uniformly distributed phase over the interval $[0; 2\pi)$. The channel is then called a *Rayleigh fading channel*, and the signal amplitude is described by the probability density function of α, which is shown in figure 2.6 and defined as

$$f_\alpha(\alpha) = \begin{cases} \frac{\alpha}{\sigma_\alpha^2} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} & \text{for } \alpha \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

where σ_α^2 is the variance of the signal amplitude.

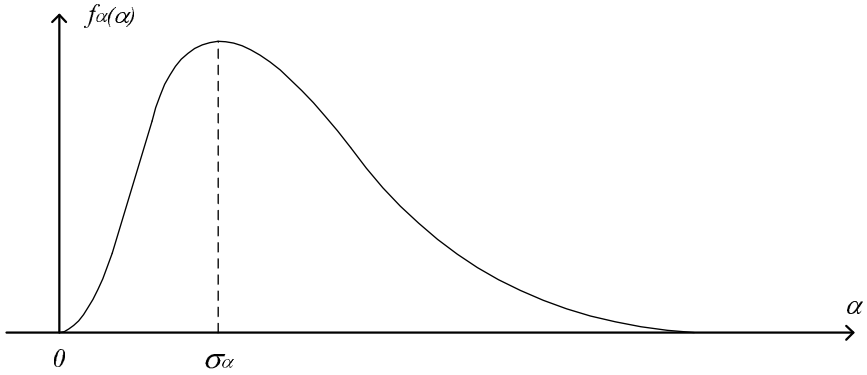


Figure 2.6: Probability density function of α.

2.2.2 Multiple Paths Fading Channel

An example of a multipath Rayleigh fading channel model with corruption of AWGN is shown as a tapped delay-line in figure 2.7. Each of the 4 paths are here characterized as a single path Rayleigh fading channel as in figure 2.5, and each tap coefficient $a_i(t)$ for $i = 1, 2, 3, 4$ is described as in equation 2.5. D denotes a fixed time-delay of τ between the paths.

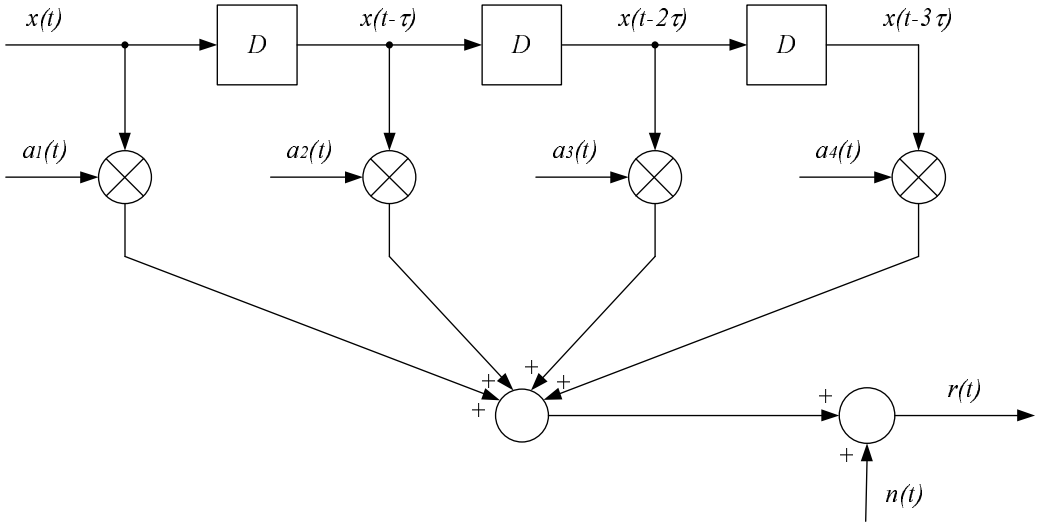


Figure 2.7: Block diagram of multipath Rayleigh fading channel with corruption of AWGN. $x(t)$ is the input signal at time t, and τ is a time-delay. The $a_i(t)$'s are the tap coeffi cients involving both amplitude and phase corruption, $n(t)$ is the white Gaussian noise process, and $r(t)$ is the output at time t.

The output of the channel depicted in figure 2.7 is expressed as

$$r(t) = \sum_{i=1}^{4} x(t-(i-1)\tau)a_i(t) + n(t) \tag{2.9}$$

$$= x(t)a_1(t) + \sum_{i=1}^{3} x(t-i\tau)a_i(t) + n(t) \tag{2.10}$$

where the middle term on the right-hand side in equation 2.10 represents the ISI.

The achievable time resolution is $1/W$, where W is the bandwidth of the transmitted signal. Assume that the time-delays in figure 2.7 is $1/W$. The number of multipath signal components is called the *multipath spread* and is denoted as T_m. In this example $T_m = 1/W \cdot 4 = 4/W$. The reciprocal of T_m is called the *coherence bandwidth* of the channel and is denoted as $B_{cb} = 1/T_m$.

If the signal bandwidth is greater than the coherence bandwidth, $(W > B_{cb})$, then the multipath components are resolvable and the frequency components of the transmitted signal are affected differently by the channel. In this case the received signal may be corrupted by ISI, and the channel is called *frequency selective*. If, on the other hand, the signal bandwidth is less than the coherence bandwidth, $(W < B_{cb})$, then the frequency components of the transmitted signal are affected similarly, and in this case the channel is called *frequency nonselective*.

The considered multipath quasi-static block fading channel is frequency selective since $W > B_{cb} = 1/T_m = W/4$, thus ISI may be produced in the channel. A block diagram of the implemented 4-path quasi-static block fading channel is illustrated in figure 2.8. The input to the channel is the k'th burst sequence $\mathbf{x}^{(k)}$, where the elements is $\mathbf{x}^{(k)}$ are denoted as $x_i^{(k)}$ for $i = 1, 2, \ldots, 114$. $a_1^{(k)}$, $a_2^{(k)}$, $a_3^{(k)}$, and $a_4^{(k)}$ are the tap coefficients for the k'th burst, and $n_i^{(k)}$ is the white Gaussian noise process realization. The output of the 4-path quasi-static block fading channel may be expressed as

$$r_i^{(k)} = x_i^{(k)} a_1^{(k)} + x_{i-1}^{(k)} a_2^{(k)} + x_{i-2}^{(k)} a_3^{(k)} + x_{i-3}^{(k)} a_4^{(k)} + n_i^{(k)} \tag{2.11}$$

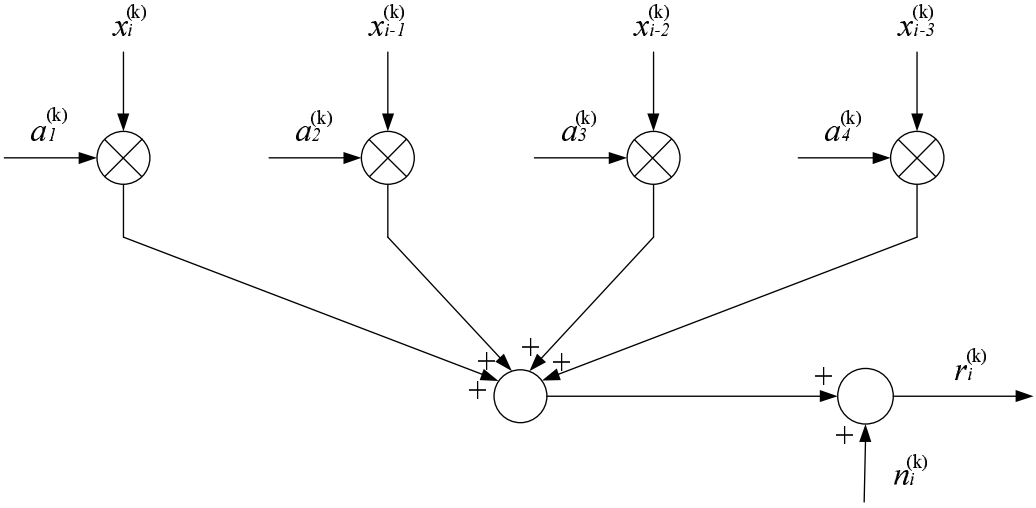


Figure 2.8: Block diagram of the implemented 4-path quasi-static block fading channel.

2.3 The Matched Filter

We consider the channel model as a linear filter with the impulse response $a(t)$ and corruption by an additive white Gaussian noise process, $n(t)$. The output of the channel, denoted as $r(t)$, is the convolution of the impulse response of the channel with the transmitted signal, $x(t)$, and corruption by AWGN

$$r(t) = a(t) * x(t) + n(t) = \int_{-\infty}^{\infty} a(\tau) x(t - \tau)\, d\tau + n(t) \tag{2.12}$$

The transmitted signal energy is spread due to the impulse response of the filter, which is of course unwanted, and an approach to minimize the affect of this spread is to employ a matched filter after the channel as shown in figure 2.9. An approach to employ the matched filter is to use a filter matched to the channel output. The filter consists of M linear filters, where each of them is matched to the channel impulse response of one of the M basis functions used for modulation. Basis functions are described in chapter 3. By doing this, the matched filter also becomes the demodulator, and this is the approach used in the implementation in this project.

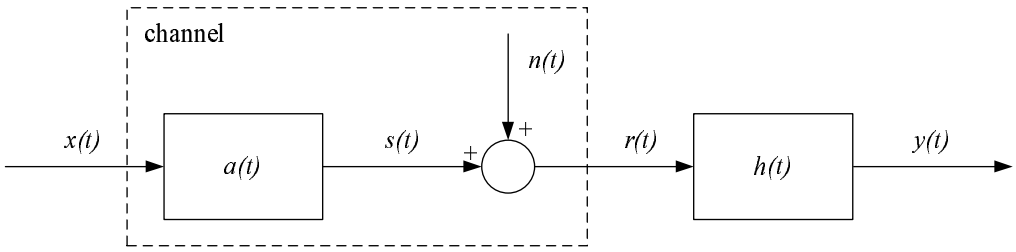


Figure 2.9: Block diagram of the channel and the matched filter with the impulse response $h(t)$. $x(t)$ is the transmitted signal, $r(t)$ is the output of the channel, and $y(t)$ is the output of the matched filter.

The output of the matched filter, $y(t)$, can be expressed as

$$\begin{aligned} y(t) &= h(t) * r(t) && (2.13)\\ &= h(t) * \big[a(t) * x(t) + n(t)\big] && (2.14)\\ &= h(t) * a(t) * x(t) + h(t) * n(t) && (2.15) \end{aligned}$$

where $h(t)$ is the impulse response of the matched filter, $a(t)$ is the impulse response of the channel, $x(t)$ is the input to the channel, and $n(t)$ is the white Gaussian noise process.

The output of the matched filter, $y(t)$, is sampled at time $t = iT$, and the result is denoted as the vector $\underline{y}$. The following derivations will only use discrete time, thus equation 2.15 can be expressed as in equation 2.16, where the matched filter is denoted as $\underline{\underline{A}}^H$. Since we have used the simple discrete-time BPSK mapping as a represention for the modulator, there is no need for a demodulator. This means that a matched filter matched to the output of the channel is the same as a filter matched to the channel, thus $h(t) = a^*(-t)$.

$$\underline{y} = \underline{\underline{A}}^H \underline{\underline{A}}\, \underline{x} + \underline{\underline{A}}^H \underline{n} \tag{2.16}$$

where

$$\underline{\underline{A}}^H = \begin{bmatrix} a_1^* & a_2^* & a_3^* & a_4^* & 0 & 0 & 0 & \dots \\ 0 & a_1^* & a_2^* & a_3^* & a_4^* & 0 & 0 & \ddots \\ 0 & 0 & a_1^* & a_2^* & a_3^* & a_4^* & 0 & \ddots \\ 0 & 0 & 0 & a_1^* & a_2^* & a_3^* & a_4^* & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix} \tag{2.17}$$

where a_1, a_2, a_3, and a_4 are the channel coefficients. $\underline{x}$ is the sampled input sequence represented as a vector, and $\underline{n}$ is the realization of $n(t)$.

The sampled output of the matched filter may also be expressed as

$$\underline{y} = \underline{\underline{R}}\,\underline{x} + \underline{\underline{A}}^H\,\underline{n} \tag{2.18}$$

where $\underline{\underline{R}} = \underline{\underline{A}}^H\underline{\underline{A}}$ is the autocorrelation matrix of the channel.

2.3.1 Properties

Most wireless channels have time-varying characteristics, and the matched filters should therefore also correspondingly change characteristics. In many communication systems, e.g. the GSM system, there is implemented a channel estimator which continuously estimates the characteristics of the channel and updates the filter so it always matches the channel or the output of the channel. The channel estimator is not a part of this project and will therefore not be implemented. Perfect channel knowledge is assumed to be known in the receiver at all time.

One important property of the matched filter is that it maximizes the *Signal-to-Noise Ratio* (SNR). To prove this property we consider the input to the matched filter, $r(t)$, as defined in equation 2.12. $|a(t)| = 1$ and therefore $x(t) = s(t)$. At the sampling instant $t = T$, the sampled output-component of the matched filter is

$$\begin{aligned} y(T) &= \int_0^T r(\tau)\cdot h(T-\tau)\,d\tau && (2.19)\\ &= \int_0^T [s(\tau)+n(\tau)]\cdot h(T-\tau)\,d\tau && (2.20)\\ &= \int_0^T s(\tau)\cdot h(T-\tau)\,d\tau + \int_0^T n(\tau)\cdot h(T-\tau)\,d\tau && (2.21)\\ &= y_s(T) + y_n(T) && (2.22) \end{aligned}$$

where $h(t)$ is the matched filter, $y_s(T)$ is the signal component, and $y_n(T)$ is the noise component.

The SNR is defined as

$$SNR = \left(\frac{S}{N}\right)_0 = \frac{y_s^2(T)}{E[y_n^2(T)]} \tag{2.23}$$

The denominator is the noise variance and may be expressed as

$$E[y_n^2(T)] = \frac{N_0}{2}\int_0^T h^2(T-t)\,dt = \sigma_n^2 \tag{2.24}$$

where σ_n^2 denotes the noise variance.

To maximize the SNR, the numerator in equation 2.23 must be maximized while the denominator is held constant. Applying the Cauchy-Schwarz inequality, the SNR can be expressed as

$$\left(\frac{S}{N}\right)_0 = \frac{y_s^2(T)}{E[y_n^2(T)]} \tag{2.25}$$

$$= \frac{\left[\int_0^T s(\tau)\cdot h(T-\tau)\,d\tau\right]^2}{\frac{N_0}{2}\int_0^T h^2(T-t)\,dt} \tag{2.26}$$

$$\leq \frac{\int_0^T s^2(\tau)\,d\tau \cdot \int_0^T h^2(T-\tau)\,d\tau}{\frac{N_0}{2}\int_0^T h^2(T-t)\,dt} \tag{2.27}$$

where the Cauchy-Schwarz inequality states that

$$\left[\int_{-\infty}^{\infty} g_1(t)g_2(t)\,dt\right]^2 \leq \int_{-\infty}^{\infty} g_1^2(t)\,dt \int_{-\infty}^{\infty} g_2^2(t)\,dt \tag{2.28}$$

and equality holds when $g_1(t) = Cg_2(t)$ for any arbitrary constant C.

According to Cauchy-Schwarz's inequality, the numerator in equation 2.26 is maximized when $s(\tau) = C\cdot h(T-\tau)$. If equality holds, the $\int_0^T h^2(T-t)\,dt$ in both the numerator and the denominator in equation 2.27 disappears, and the maximum SNR of the matched filter is

$$\left(\frac{S}{N}\right)_0 = \frac{2}{N_0}\int_0^T s^2(t)\,dt \tag{2.29}$$

$$= \frac{2E_s}{N_0} \tag{2.30}$$

where E_s is the energy in each symbol after encoding.

CHAPTER 3

Modulation

The purpose of a communication system is to deliver information to a user at the destination. To do this, a *message signal* is created based on the information, and it is modified by a transmitter into a signal suitable for transmission through the channel. The modification is achieved by means of a *modulation process*, which varies some parameters of a carrier wave in accordance with the message signal. The receiver re-creates a degraded version of the original message signal by using a *demodulation process*, which is the reverse of the modulation process used in the transmitter. The reason for degration of the re-created message signal is due to the unavoidable presence of noise and distortion in the received signal, and this resulting degration is influenced by the type of modulation scheme used.

The modulation process can be classified into *continuous-wave modulation* and *pulse modulation*. For continuous-wave modulation, the carrier wave is a sinusoidal wave, and for pulse modulation the carrier is a periodic sequence af rectangular pulses.

We consider discrete time BPSK modulation represented as a baseband signal. The input to the modulator is burst sequences in $\{0, 1\}$. The BPSK modulator (or mapper) maps each element in the bursts sequences into a symbol in $\{-1, +1\}$.

To give a general understanding of how BPSK works, both the BPSK modulation for bandpass transmission and the corresponding representation for baseband transmission are described in this chapter.

Consider a communication system, where the transmitted symbols belong to an alphabet of M symbols denoted as $m_1, m_2, \ldots, m_M$. In the case for BPSK, the alphabet consists of two symbols, thus the name *Binary* PSK. The transmitter codes for each duration T the symbol m_i into a distinct signal, $s_i(t)$, suitable for transmission through the channel. $s_i(t)$ is a real-valued signal waveform, and it occupies the full duration allotted to m_i. The M signal waveforms, $s_1(t), s_2(t), \ldots, s_M(t)$, can be represented as geometric representations, where they are linear combinations of K orthonormal waveforms for $K \leq M$. The procedure to construct the orthonormal waveforms is called the *Gram-Schmidt orthogonalization procedure*, and it is described in the first section of this chapter.

The references for this chapter are [Proakis et al., 2002] and [Haykin, 2001].

3.1 Gram-Schmidt Orthogonalization Procedure

Consider the set of M signal waveforms

$$\mathbb{S} = \{s_1(t), s_2(t), \ldots, s_M(t)\} \tag{3.1}$$

From the M signal waveforms in $\mathbb{S}$ we construct $K \leq M$ orthonormal waveforms

$$\mathbb{S}_\psi = \{\psi_1(t), \psi_2(t), \ldots, \psi_K(t)\} \tag{3.2}$$

The k'th orthonormal waveform is defined as

$$\psi_k(t) = \frac{{s'}_k(t)}{\sqrt{E_k}} \tag{3.3}$$

where

$$s'_k(t) = s_k(t) - \sum_{i=1}^{k-1} c_{k,i}\,\psi_i(t) \tag{3.4}$$

and

$$E_k = \int_{-\infty}^{\infty} |s'_k(t)|^2\,dt \tag{3.5}$$

$\psi_k(t)$ is a normalization of $s'_k(t)$ to unit energy. $c_{k,i}$ is a projection of $s_k(t)$ onto $\psi_i(t)$, and $s'_k(t)$ is therefore orthogonal to $\psi_i(t)$ (but not with unit energy).

The projection, $c_{k,i}$, is defined as

$$c_{k,i} = \langle s_k(t), \psi_i(t)\rangle = \int_{-\infty}^{\infty} s_k(t)\psi_i(t)\,dt \qquad \text{for } i = 1, 2, \ldots, k-1 \tag{3.6}$$

When the K orthonormal waveforms in $\mathbb{S}_\psi$ are constructed, the M signal waveforms in $\mathbb{S}$ can be expressed as linear combinations of the orthonormal waveforms

$$s_m(t) = \sum_{k=1}^{K} \theta_{m,k}\psi_k(t) \qquad \text{for } m = 1, 2, \ldots, M \tag{3.7}$$

where

$$\theta_{m,k} = \int_{-\infty}^{\infty} s_m(t)\psi_k(t)\,dt \tag{3.8}$$

The energy in each signal waveform, $s_m(t)$, is the sum of the energies in the K θ's

$$E_m = \int_{-\infty}^{\infty} |s_m(t)|^2\,dt = \sum_{k=1}^{K} \theta_{m,k}^2 \tag{3.9}$$

The signal waveforms may be expressed as a point in the K dimensional signal space, and the K θ's are the coordinates

$$s_m = (\theta_{m,1}, \theta_{m,2}, \ldots, \theta_{m,K}) \tag{3.10}$$

It should be noted, that by applying the Gram-Schmidt orthogonalization procedure, you are guaranteed to find a set of orhonormal waveforms for the signal waveforms, but in many cases it is simpler to construct the orthonormal functions just by inspection.

3.2 Binary Phase Shift Keying

Phase Shift Keying (PSK) is a continuous-wave modulation scheme for bandpass transmission, and it consists of M signal waveforms, $s_1(t), s_2(t), \ldots, s_M(t)$. The signals in the M-ary PSK modulation scheme can be represented by 2 orthonormal waveforms, $\psi_1(t)$ and $\psi_2(t)$, and by expressing the M signal waveforms as points in a 2-dimensional signal space according to equation 3.10, we have $s_m = (\theta_{m,1}, \theta_{m,2})$. The PSK signal constellation for $M = 2, 4, 8$ is shown in figure 3.1.

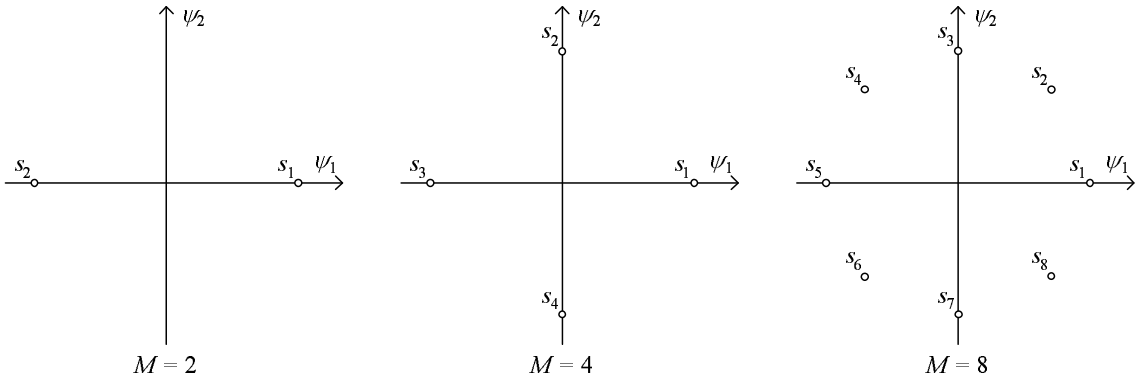


Figure 3.1: PSK signal constellation for $M = 2, 4, 8$.

3.2.1 Bandpass Transmission

PSK consists of sine waveforms with the same frequency, and the information is conveyed by the phase. The signal waveforms for M-ary PSK are defined as

$$s_m(t) = g_T(t)\cos(2\pi f_c t + \phi_m) \qquad \text{for } t \in [0, T] \tag{3.11}$$

where

$$g_T(t) = \sqrt{\frac{2E_s}{T}} \qquad \text{for } t \in [0, T] \tag{3.12}$$

$$\text{and} \qquad \phi_m = 2\pi\frac{m-1}{M} \qquad \text{for } m = 1, 2, \ldots, M \tag{3.13}$$

$g_T(t)$ is a rectangular pulse, E_s is the transmitted signal energy per symbol, f_c is the carrier frequency, T is the symbol duration time, and ϕ_m is the phase. f_c is chosen so that $f_c = \frac{C}{T}$ for some fixed integer C, and T is positive.

The signal waveforms defined in equation 3.11 can be split up into two signals by viewing the angle of the cosine as a sum of two angles. Equation 3.11 can then be rewritten as

$$s_m(t) = \sqrt{\frac{2E_s}{T}}\left[A_1 \cos(2\pi f_c t) - A_2 \sin(2\pi f_c t)\right] \qquad \text{for } t \in [0, T] \tag{3.14}$$

where

$$A_1 = \cos\left(2\pi \frac{m-1}{M}\right) \qquad \text{for } m = 1, 2, \ldots, M \tag{3.15}$$

$$\text{and} \qquad A_2 = \sin\left(2\pi \frac{m-1}{M}\right) \qquad \text{for } m = 1, 2, \ldots, M \tag{3.16}$$

The 2-dimensional geometric point-representation stated in equation 3.10 can then be expressed as

$$s_m = (\theta_{m,1}, \theta_{m,2}) = \left(\sqrt{E_s}\cos\left(2\pi \frac{m-1}{M}\right), \sqrt{E_s}\sin\left(2\pi \frac{m-1}{M}\right)\right) \tag{3.17}$$

For BPSK, the modulation scheme has two signal waveforms $s_1(t)$ and $s_2(t)$, where $\phi_1 = 0$ and $\phi_2 = \pi$, or $s_2(t) = -s_1(t)$, and the signal constellation for has only one orthonormal waveform

$$\psi_1(t) = \sqrt{\frac{2}{T}}\cos(2\pi f_c t) \qquad \text{for } t \in [0, T]. \tag{3.18}$$

The input to the modulator is a burst sequence, denoted as **e**, where the elements in **e** are denoted as $e_i \in \{0, 1\}$. The transmitted signal after modulation, denoted as $x(t)$, is expressed as

$$x(t) = \begin{cases} s_1(t) & \text{for } e_i = 0 \\ s_2(t) & \text{for } e_i = 1 \end{cases} \tag{3.19}$$

where $s_1(t) = +\sqrt{E_S} \cdot \psi_1(t)$ and $s_2(t) = -\sqrt{E_S} \cdot \psi_1(t)$.

A BPSK modulated signal with $f_c = \frac{1}{T}$ for the sequence $\{0, 0, 1\}$ is shown in figure 3.2.

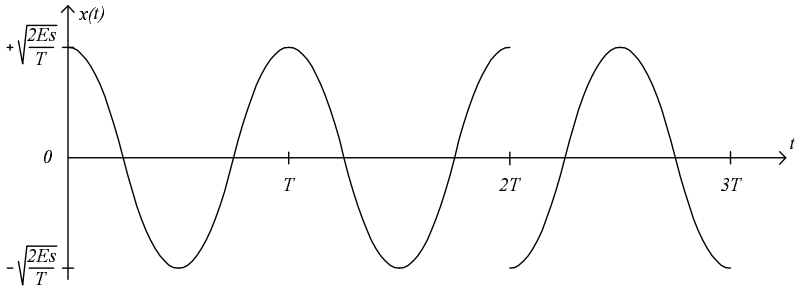


Figure 3.2: A BPSK modulated signal with $f_c = \frac{1}{T}$ for the sequence $\{0, 0, 1\}$.

3.2.2 Coherent Demodulation

It has now been described how a burst sequence, **e**, is mapped into a waveform, $x(t)$. After $x(t)$ is transmitted through a channel, the channel corrupted signal, denoted as $r(t)$, is received and demodulated into a burst sequence, denoted as $\mathbf{y}$.

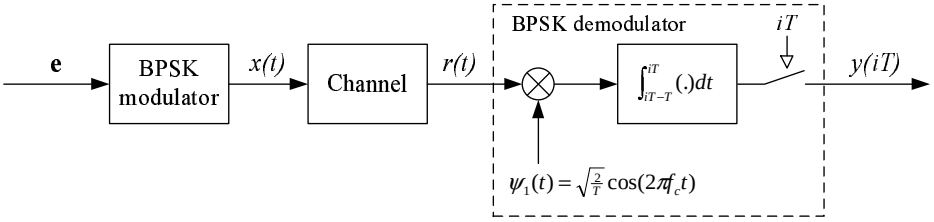


Figure 3.3: A bandpass BPSK modulation scheme.

The system with BPSK modulation scheme is shown in figure 3.3. The channel corrupted received signal, $r(t)$, is multiplied with the orthonormal function for BPSK, $\psi_1(t)$, and the product is integrated over the time duration T. The output of the integrator is sampled at each time instant iT and is also the output of the demodulator, $y(iT)$.

We assume that $x(t)$ is transmitted through an AWGN channel, and the received signal, $r(t)$, is corrupted by AWGN. $r(t)$ is the input to the demodulator, and the sampled output for the first symbol duration is computed as

$$y(T) = \int_0^T r(t)\psi_1(t)\,dt \tag{3.20}$$

$$= \int_0^T [x(t) + n(t)]\sqrt{\frac{2}{T}}\cos(2\pi f_c t)\,dt \tag{3.21}$$

where $n(t)$ is the contribution due to the corruption by the noise in the channel.

If we replace the contribution due to the channel corruption with $n_T = \int_0^T n(t)\sqrt{\frac{2}{T}}\cos(2\pi f_c t)\,dt$, the expression in equation 3.21 can be rewritten as

$$y(T) = \int_0^T x(t)\sqrt{\frac{2}{T}}\cos(2\pi f_c t)\,dt + n_T \tag{3.22}$$

$$= \int_0^T \sqrt{\frac{2E_s}{T}}\cos(2\pi f_c t + \phi_m)\sqrt{\frac{2}{T}}\cos(2\pi f_c t)\,dt + n_T \tag{3.23}$$

$$= \int_0^T \sqrt{\frac{2E_s}{T}}\sqrt{\frac{2}{T}}\left[\frac{1}{2}\cos(4\pi f_c t + \phi_m) + \frac{1}{2}\cos(\phi_m)\right]dt + n_T \tag{3.24}$$

$$= \frac{1}{2}\sqrt{\frac{4E_s}{T^2}}\int_0^T \left[\cos(4\pi f_c t + \phi_m) + \cos(\phi_m)\right]dt + n_T \tag{3.25}$$

$$= \frac{\sqrt{E_s}}{T}\left[\sin(4\pi f_c T + \phi_m) - \sin(\phi_m) + T\cos(\phi_m)\right] + n_T \tag{3.26}$$

Since f_cT is a fixed integer, C, the expression in equation 3.26 can be reduced to

$$y(T) = \frac{\sqrt{E_s}}{T}\big[\sin(\phi_m) - \sin(\phi_m) + T\cos(\phi_m)\big] + n_T \tag{3.27}$$

$$= \sqrt{E_s}\cos(\phi_m) + n_T \tag{3.28}$$

and for BPSK, ϕ_m is either 0 or π, therefore

$$y(T) = \begin{cases} +\sqrt{E_s} + n_T & \text{for } \phi_m = 0 \\ -\sqrt{E_s} + n_T & \text{for } \phi_m = \pi \end{cases} \tag{3.29}$$

3.2.3 Baseband Transmission

For baseband transmission, the carrier wave is omitted, and only the message signal is considered. The message signal for BPSK is represented in the form of a discrete pulse-amplitude modulated signal and transmitted directly over a low-pass channel as a periodic sequence af rectangular pulses. This modulation scheme is called *Pulse-Amplitude Modulation* (PAM), and an example is shown in figure 3.4.

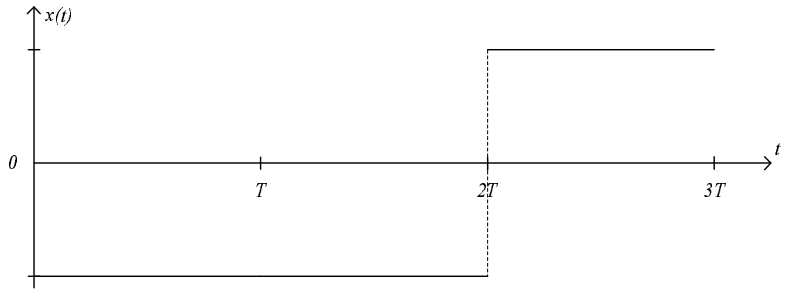


Figure 3.4: Example of a baseband transmitted discrete pulse-amplitude modulated signal for the sequence {0,0,1}.

In this project, the BPSK modulated signal is represented as a discrete time baseband signal, and therefore are all the signals considered as sequences as shown in figure 3.5.

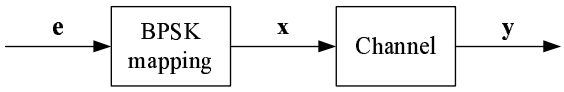


Figure 3.5: A discrete time system model with BPSK modulation scheme.

The two element-values in **e** (0 and 1) can be represented as vectors with phase shifted 180 degrees or π. This gives the geometrical representations $s_1 = +\sqrt{E_s}$ and $s_2 = -s_1 = -\sqrt{E_s}$. Therefore should the BPSK modulator simply map the sequence **e** $\in \{0, 1\}$ into the sequence **x** $\in \{-\sqrt{E_s}, +\sqrt{E_s}\}$. Note that $\sqrt{E_s} = 1$.

3.2.4 Error Probability

Since the transmitted signal is corrupted by the channel, there is a probability that hard decision on a observed symbol results in a wrong bit. the AWGN channel is considered, an error occur when

$y_i < 0$ is observed when $x_i = +\sqrt{E_s}$ is sent, or when $y_i > 0$ is observed when $x_i = -\sqrt{E_s}$ is sent.

The conditional probability density function for y_i given that $x_i = +\sqrt{E_s}$ is defined as [Haykin, 2001]

$$f_{Y|X}(y_i|x_i = +\sqrt{E_s}) = \frac{1}{\sqrt{\pi N_0}} e^{-\frac{1}{N_0}|y_i - \sqrt{E_s}|^2} \tag{3.30}$$

The conditional symbol error-probability is then defined as

$$\begin{aligned} P(y_i < 0|x_i = +\sqrt{E_s}) &= \int_{-\infty}^{0} f_{Y|X}(y_i| + \sqrt{E_s})\, dy_i & (3.31)\\ &= \frac{1}{\sqrt{\pi N_0}} \int_{-\infty}^{0} e^{-\frac{1}{N_0}|y_i - \sqrt{E_s}|^2}\, dy_i & (3.32)\\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{-\sqrt{2E_s/N_0}} e^{-\frac{1}{2}z^2}\, dz & (3.33) \end{aligned}$$

where

$$z = \frac{y_i - \sqrt{E_s}}{\sqrt{\frac{N_0}{2}}} \tag{3.34}$$

The conditional symbol error-probability in equation 3.33 can be rewritten as

$$\begin{aligned} P_s &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{-\sqrt{2E_s/N_0}} e^{-\frac{1}{2}z^2}\, dz & (3.35)\\ &= \int_{-\infty}^{-\sqrt{2E_s/N_0}} q(z)\, dz & (3.36)\\ &= Q\left(\sqrt{\frac{2E_s}{N_0}}\right) & (3.37) \end{aligned}$$

where

$$q(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z^2} \qquad \text{and} \qquad Q(v) = \int_{v}^{\infty} q(z)\, dz \tag{3.38}$$

For BPSK the symbol error-probability is equal to the bit-error propability and therefore

$$P_b = Q\left(\sqrt{\frac{2E_b}{N_0}}\right) = Q\left(\sqrt{\frac{2E_s}{N_0}}\right) = P_s \tag{3.39}$$

The bit-error propability, or *Bit Error Rate* (BER), decreases for increasing SNR. Simulations of transmission through the AWGN channel with BPSK is shown in the BER plot in figure 3.6. The simulation results is compared with the theoretical lower bound for transmission through the AWGN channel with BPSK modulation scheme, which is defined is equation 3.39.

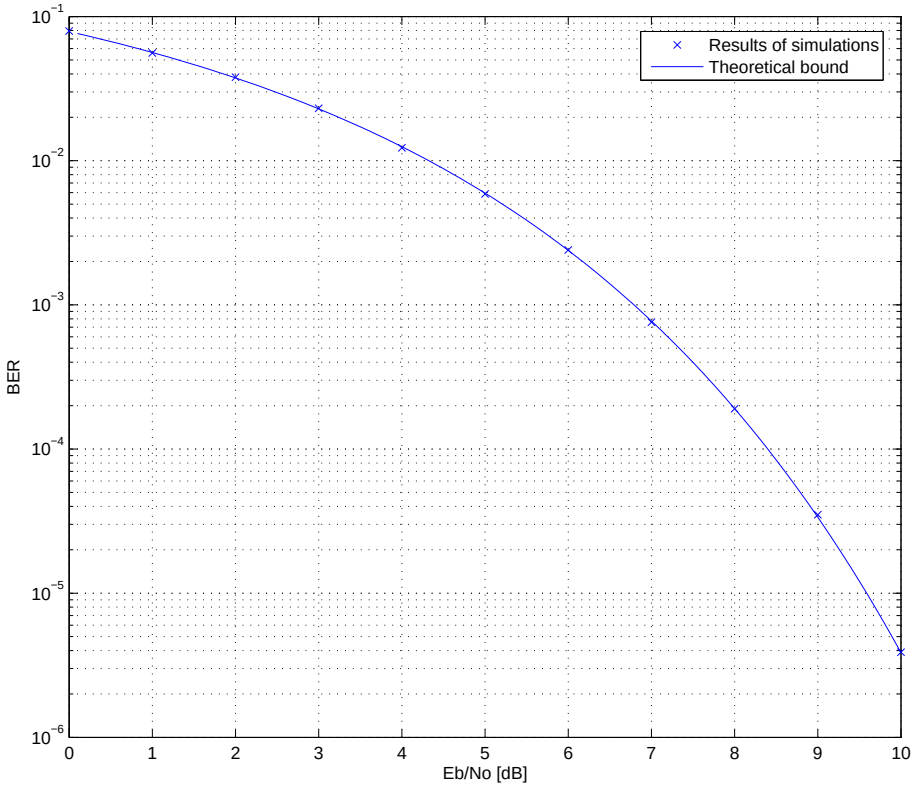


Figure 3.6: BER plot of the AWGN channel with BPSK.

CHAPTER 4

Convolutional Encoding

The system model consider (23,33) convolutional coding as channel coding, since it is the scheme specified in the GSM standard. The input to the encoder is the info word, which is a sequence of 224 info bits, and the output is the code word consisting of 456 code bits.

The purpose of channel coding is to provide reliable transmission of information in the presence of noise by adding redundancy to the transmitted signal in a controlled manner. In general, the channel encoder separates or segments an incoming bit stream into equal length blocks of k binary digits and maps each k-bit block, which is called the info word, into an l-bit code word block, where $l > k$. The set of code words contains 2^k code words of length l code bits. After transmission and detection in the receiver, the channel decoder finds the most likely code word from the received l-bit word, and inversely maps it into the corresponding k-bit info word.

In this project, convolutional block codes have been studied. Block codes process the info bits on a block-by-block basis, so as to treat each block of info bits independently from each other. The block codes map k-bit info words into l-bit code words, which imply that $l - k$ additional bits are added to the k-bit info word to form the l coded bits.

This chapter will describe the convolutional encoder with emphasis on the (23,33) convolutional encoder, which is specified in the GSM standard. The primary reference of this chapter is [Proakis et al., 2002].

4.1 The Convolutional Encoder

For convolutional codes the encoded code bits depend not only on the current k info bits but also on the previous m info bits.

Rate 1/2 convolutional codes have a simple encoding scheme. The k info bits are passed sequentially to m linear shift registers, which are basically delay elements, and these k info bits are mapped into l code bits. These code bits are formed from modulo-2 additions of the contents of the memory (m-shift registers) and the info bits. After the k info bits are passed, m zeros are also passed through the encoder so as to reset the encoder to all-zero state and make it ready for next transmission. The code rate of binary convolutional encoder is $R_c = k/l$.

Convolutional codes are specified by there *generators* denoted as g_1 and g_2, and the specific code is called a *(g_1,g_2) convolutional code*. The generators have an octal value which define the mapping process from the k info bits to the l code bits. An example of a convolutional encoder having generators $(5, 7)$ with code rate $R_c = 1/2$ is shown in figure 4.1. The input is the info word sequence denoted as **d**, where the elements in **d** is denoted as d_i for $i = 1, 2, \ldots, k$. The output is the code word sequence, denoted as **c**, where the elements in **c** is denoted as $\underline{c}_i$ for $i = 1, 2, \ldots, k$, and $\underline{c}_i$ consist of two elements, $c_i^{(1)}$ and $c_i^{(2)}$, so the code word sequence contains $l = 2k$ elements.

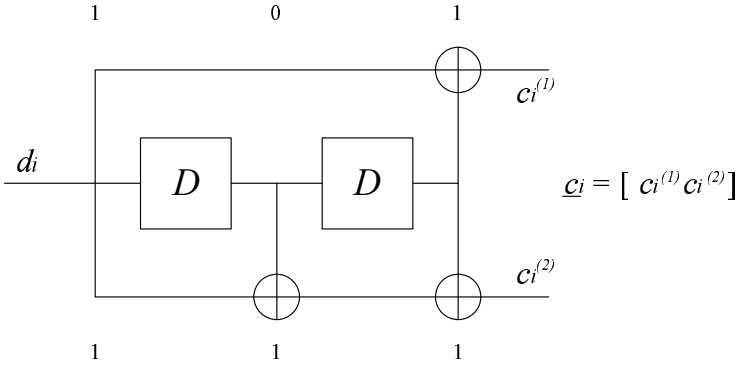


Figure 4.1: (5,7) convolutional encoder. d_i is the uncoded 1-dimensional info word element, and $\underline{c}_i$ is the 2-dimensional code word element. D denotes a time-delay. The binary values for the generators, $g_1 = 5$ and $g_2 = 7$, is 101 and 111 respectively.

Example 4.1 will show the mapping from the info word sequence **d** into the code word sequence **c** with the (5,7) generators and code rate $R_c = 1/2$.

EXAMPLE 4.1: The sequence $\mathbf{d} = \{1, 0, 1, 0, 0\}$ is encoded with a $\frac{1}{2}$ rate (5,7) convolutional encoder. It is assumed that the encoder is in all zero state before encoding. $\oplus$ denotes modulo-2 addition.

The elements in $\mathbf{c}^{(1)}$ and $\mathbf{c}^{(2)}$ are calculated as

$$\begin{aligned}
\mathbf{c}^{(1)} &= \{d_1 \oplus 0 \ , \ d_2 \oplus 0 \ , \ d_3 \oplus d_1 \ , \ d_4 \oplus d_2 \ , \ d_5 \oplus d_3\} && (4.1)\\
&= \{1, 0, 0, 0, 1\} && (4.2)\\
\mathbf{c}^{(2)} &= \{d_1 \oplus 0 \oplus 0 \ , \ d_2 \oplus d_1 \oplus 0 \ , \ d_3 \oplus d_2 \oplus d_1 \ , && (4.3)\\
&\qquad d_4 \oplus d_3 \oplus d_2 \ , \ d_5 \oplus d_4 \oplus d_3\}\\
&= \{1, 1, 0, 1, 1\} && (4.4)
\end{aligned}$$

The elements in **c** is then calculated as

$$\begin{aligned}
\mathbf{c} &= \{c_1^{(1)}c_1^{(2)} \ , \ c_2^{(1)}c_2^{(2)} \ , \ c_3^{(1)}c_3^{(2)} \ , \ c_4^{(1)}c_4^{(2)} \ , \ c_5^{(1)}c_5^{(2)}\} && (4.5)\\
&= \{11, 01, 00, 01, 11\} && (4.6)
\end{aligned}$$

where $c_i^{(1)}$ and $c_i^{(2)}$ are the elements in $\underline{\mathbf{c}}_i$ for $i = 1, 2, ..., 5$.

The GSM encoder has the generators $g_1 = 23$ and $g_2 = 33$ and code rate $R_c = 1/2$ as illustrated in figure 4.2. The code word is computed the same way as depicted in example 4.1, except that $c_i^{(1)}$ and $c_i^{(2)}$ is now calculated as

$$\begin{aligned}
c_i^{(1)} &= d_i \oplus d_{i-3} \oplus d_{i-4} && (4.7)\\
c_i^{(2)} &= d_i \oplus d_{i-1} \oplus d_{i-3} \oplus d_{i-4} && (4.8)
\end{aligned}$$

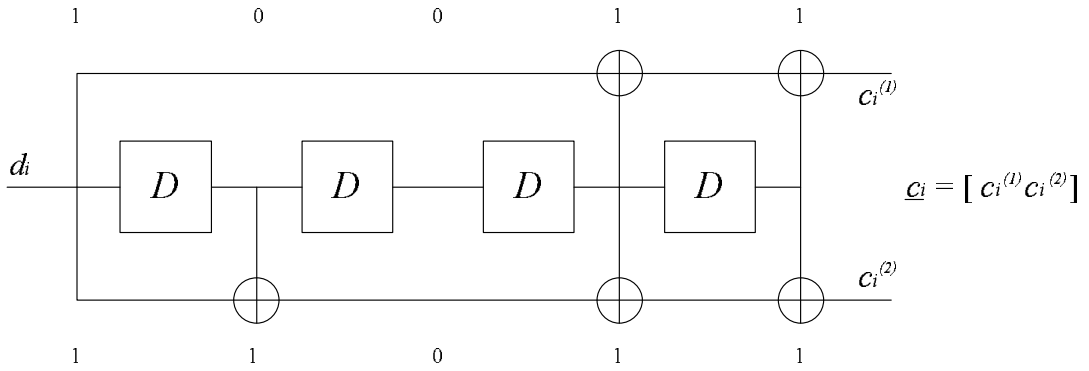


Figure 4.2: (23,33) convolutional encoder. The binary values for $g_1 = 23$ and $g_2 = 33$ is 10011 and 11011 respectively, as depicted in the fi gure.

In the implemented system, an info word, **d**, consisting of 224 info bits is to be encoded using the (23,33) encoding scheme. Before encoding, 4 zeros are added to **d** to ensure zero-state. The resulting code word, **c**, has the size of $(224 + 4) \cdot 2 = 456$ symbols.

The elements in **c** are computed as

$$c_k = \begin{cases} d_{k_o} \oplus d_{k_o-3} \oplus d_{k_o-4} & \text{for } k \text{ is odd} \\ d_{k_e} \oplus d_{k_e-1} \oplus d_{k_e-3} \oplus d_{k_e-4} & \text{for } k \text{ is even} \end{cases} \tag{4.9}$$

for $k = 1, 2, \ldots, 456$. $k_o = \frac{k+1}{2}$ and $k_e = \frac{k}{2}$.

4.2 Properties of Convolutional Codes

Convolutional codes can be regarded as a finite state machine, represented by a state diagram. Each of the 2^m states in the state diagram, where m is the number of memory elements, is represented by a circle and transitions among the states are represented by lines connecting these circles. On every line, input and output caused by that transition are represented. A state diagram for the convolutional encoder having generators (5,7) is shown in figure 4.3. Dashed line denotes '0'-input and solid line denotes '1'-input.

Convolutional codes may also be described by a trellis diagram. The state diagram fails to represent the transitions as time evolves. The trellis diagram is able to represent the transitions among the various states as the time evolves. This is shown by repeating the states vertically along the time axis (horizontal discrete axis). See figure 4.4. Every repetition of the states is called a stage, and the step from one state to the next is called a transition (or branch). The transition from one state to another in the next stage is represented by a dashed line for '0'-input and solid line for '1'-input. In other words, it can be said that the trellis diagram is a repetition of the state diagram along the time axis.

The trellis diagram contains several paths, each path corresponding to a distinct code word. An example of a trellis diagram for (5,7) convolutional codes is shown in figure 4.4. It is assumed that the encoder is in all zero state before encoding, thus has the first stage only transitions from state '00'. The two digits connected to each transition is the output of the encoder corresponding to the transition.

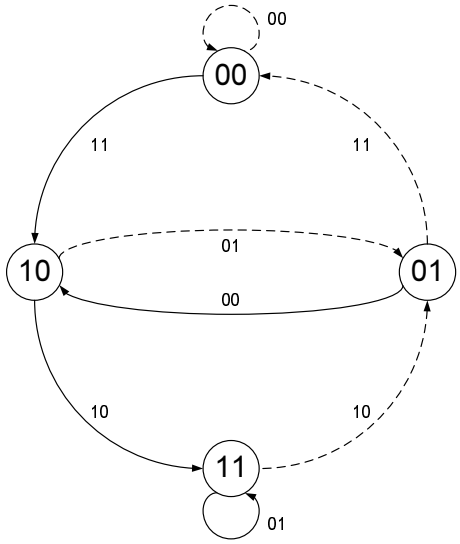


Figure 4.3: State diagram of a (5,7) convolutional encoder. Dashed line denotes '0'-input and solid line denotes '1'-input.

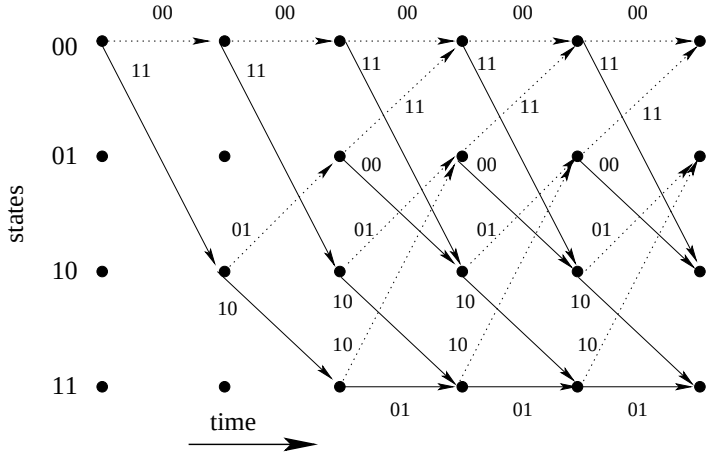


Figure 4.4: Trellis diagram of a (5,7) convolutional encoder. Dashed line denotes '0'-input and solid line denotes '1'-input.

The trellis diagram for (23,33) convolutional codes has $2^m = 2^4 = 16$ states as depicted in figure 4.5. The figure shows only one transition.

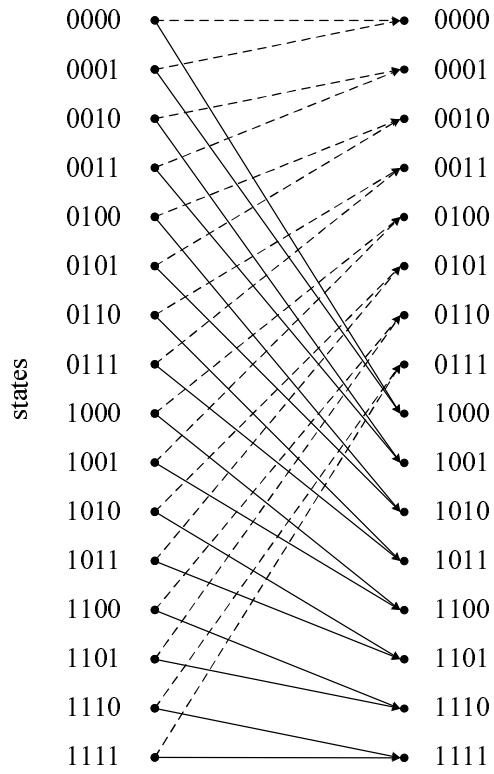


Figure 4.5: Trellis diagram of a (23,33) convolutional encoder. Dashed line denotes '0'-input and solid line denotes '1'-input.

CHAPTER 5

Convolutional Decoding using the Viterbi Algorithm

The decoder maps the observed code word of 456 code bits into an estimated info word of 224 info bits. The GSM standard does not specify any decoding algorithm, but an often used channel decoder for convolutional coding is a *Maximum Likelihood* (ML) decoder called the Viterbi Decoder. The basic idea of ML-decoding is to compare the received sequence with all the possible code word sequences and then calculate the corresponding likelihoods. The sequence with the highest likelihood is selected as the estimated code word, which is then mapped into the estimated info word.

The Viterbi Algorithm and the performance of the Viterbi Decoder is described in this chapter. The reference for the chapter is [Proakis et al., 2002], unless any other is stated.

5.1 Maximum Conditional Probability

The Viterbi Decoder finds the closest code word to the observed sequence by processing the sequences on a bit-by-bit basis. This means that the Viterbi Algorithm selects the code word that maximizes the conditional probability. [Morelos-Zaragoza, 2002]

The conditional probability for a code word **c**, given the observed sequence **y** is

$$P(\mathbf{c}|\mathbf{y}) = \prod_{j=l}^{l} P(c_j|y_{2j-1}, y_{2j}) \tag{5.1}$$

where y_j is the j'th element in **y**. c_j is the j'th element in **c**, and l is the length of the code word.

The estimated code word $\hat{\mathbf{c}}$ can then be expressed as

$$\hat{\mathbf{c}} = \arg\max_{\mathbf{c}\in\mathbb{C}} P(\mathbf{c}|\mathbf{y}) = \arg\max_{\mathbf{c}\in\mathbb{C}} \prod_{j=l}^{l} P(c_j|y_j) \tag{5.2}$$

where $\mathbb{C}$ is the set of code words.

For the AWGN channel it is equivalent to choosing the code word that minimizes the squared Euclidean distance

$$\hat{\mathbf{c}} = \arg\min_{\mathbf{c}\in\mathbb{C}} d_E(\mathbf{c},\mathbf{y}) = \arg\min_{\mathbf{c}\in\mathbb{C}} \sum_{j=1}^{l} \|c_j - y_j\|^2 \tag{5.3}$$

When using the squared Euclidean distance, it is called soft-decision decoding. Another way to make decisions is to use the hamming distance, and this is called hard-decision decoding.

The hamming distance is defined as

$$d_H(\mathbf{c},\mathbf{y}') = \sum_{j=1}^{l} c_j \oplus {y'}_j \tag{5.4}$$

where $\mathbf{y}$ is turned into a binary sequence, $\mathbf{y}$, by making hard-decisions on the individual elements in $\mathbf{y}$.

The output of the Viterbi decoder is an estimated info word $\widehat{\mathbf{d}}$, corresponding to the estimated code word $\widehat{\mathbf{c}}$.

5.2 Optimal Path Through the Trellis

Convolutional codes can be represented by a trellis diagram as described in chapter 4. The Viterbi Decoder searches for the path through the trellis that is at minimum distance from received sequence. This indicates the most likely transmitted code word, and thereby the most likely info word corresponding to that code word.

To describe how the Viterbi Decoder works, we will follow example 5.1, where hard-decisions decoding is used on a (5,7) convolutional encoded sequence with code rate $R_c = 1/2$.

EXAMPLE 5.1: The sequence $\mathbf{y}' = \{11, 11, 01, 01, 10\}$ is hard-decisions of the observed sequence $\mathbf{y}$, and the transmitted code word is encoded with a $\frac{1}{2}$ rate (5,7) convolutional encoder as in example 4.1 on page 31. $\mathbf{y}'$ has three errors compared to the code word in example 4.1, where $\mathbf{c} = \{11, 01, 00, 01, 11\}$. It is assumed that the encoder was in all zero state before encoding.

Each node in the trellis has a state-value, denoted as $S_i^{(k)}$, corresponding to the path with the shortest hamming distance to that node. The transition to this node with the shortest Hamming distance is called the "survivor", and the other is discarded. The state-values are gathered in a state matrix, denoted as $M(S_i^{(k)})$, as shown in table 5.1.

$k\backslash i$	0	1	2	3	4	5
1 ('00')	$S_0^{(1)}$	$S_1^{(1)}$	$S_2^{(1)}$	$S_3^{(1)}$	$S_4^{(1)}$	$S_5^{(1)}$
2 ('01')	$S_0^{(2)}$	$S_1^{(2)}$	$S_2^{(2)}$	$S_3^{(2)}$	$S_4^{(2)}$	$S_5^{(2)}$
3 ('10')	$S_0^{(3)}$	$S_1^{(3)}$	$S_2^{(3)}$	$S_3^{(3)}$	$S_4^{(3)}$	$S_5^{(3)}$
4 ('11')	$S_0^{(4)}$	$S_1^{(4)}$	$S_2^{(4)}$	$S_3^{(4)}$	$S_4^{(4)}$	$S_5^{(4)}$

Table 5.1: State matrix, $M(S_i^{(k)})$, where k denotes the state and i denotes the stage. $S_0^{(k)}$ is the initial state-values for stage k.

The state-values are calculated by the following algorithm

$$S_i^{(k)} = S_{i-1}^{(j)} + d_H(y_i', c_i) \qquad \text{for } i = 1, 2, \ldots, 5 \quad \text{and} \quad k = 1, 2, 3, 4 \tag{5.5}$$

where $S_{i-1}^{(j)}$ is the previous state-value in the survivor path. Since the encoder was in all zero state before encoding, the initial state-values are

$$S_0^{(k)} = \quad 0 \qquad \text{for } k = 1 \tag{5.6}$$

$$S_0^{(k)} = \quad \infty \qquad \text{for } k = 2, 3, 4 \tag{5.7}$$

All the necessary state-values are calculated and placed in the trellis diagram in figure 5.1. The trellis ends in '00'-state, because two zeros have been added in the end of the sequence in the encoder in order to get the encoder in all zero state after encoding. The crosses denote discarded transitions. As shown in the figure, the final survivor-path, denoted as a blue path, indicates that the most likely code word was $\{11, 01, 00, 01, 11\}$, which corresponds to the info word $\{1, 0, 1, 0, 0\}$, which also was the case.

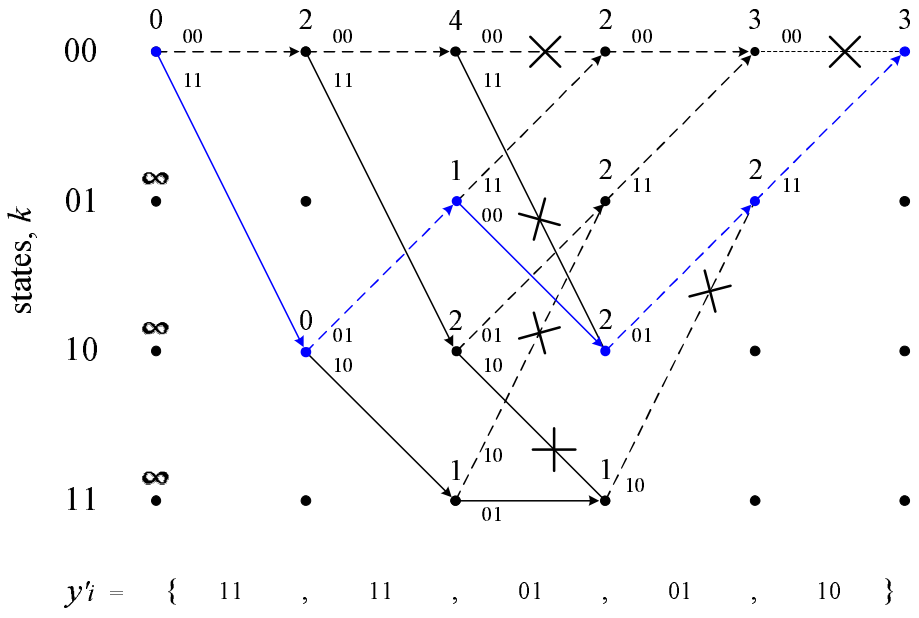


Figure 5.1: The trellis diagram used as hard-decision Viterbi decoding. The dashed lines correspond to '0'-input and the solid lines to '1'-input. The numbers above the state-points is the state-values, and the blue path is the final survivor-path, and it corresponds to the info word $\{1, 0, 1, 0, 0\}$.

The decoding technique for the (23,33) convolutional decoder is the same as for (5,7), except that the trellis diagram has 16 states in stead of 4, which is explained in chapter 4 and depicted in figure 4.5.

5.3 Performance

The Viterbi Decoder works for both soft- and hard-decision decoding. Simulations of transmission through the AWGN channel with BPSK and with (5,7) convolutional coding hard- and soft-decision using the Viterbi Decoder is shown in figure 5.2 to compare the performances. The difference between the hard- and soft-decision Viterbi decoders is approximately 2 dB at $BER = 10^{-3}$. It is thereby clear that the soft-decision decoding scheme has better performance than the hard-decision decoding scheme. The figure also shows the theoretical bound, defined in chapter 3, which is the lower bound for transmission through the AWGN channel with BPSK modulation scheme and no coding.

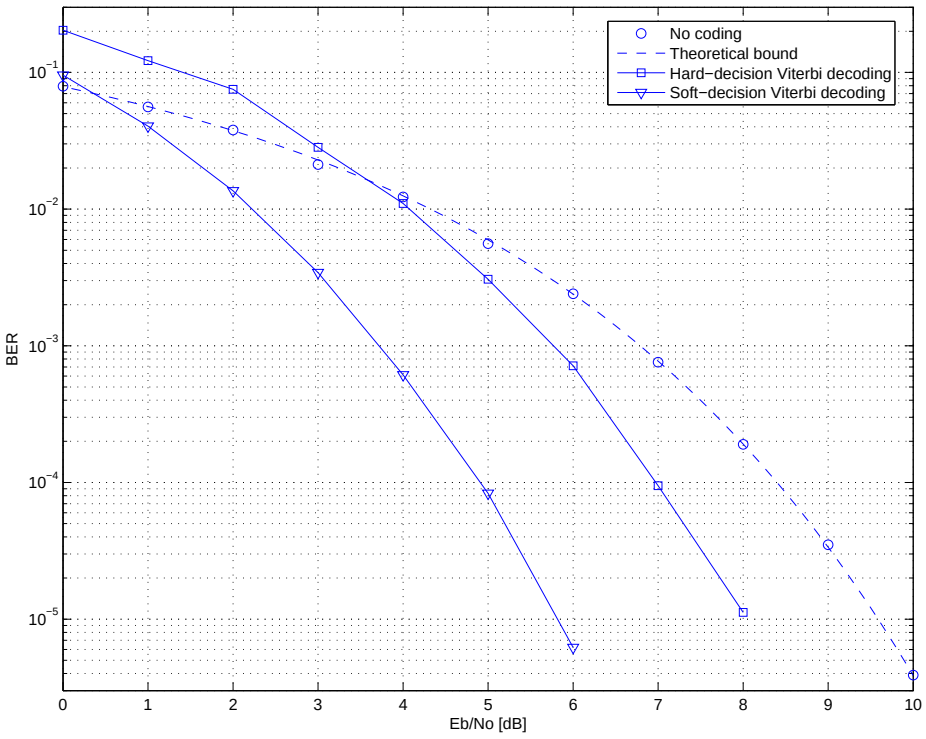


Figure 5.2: BER plot for transmission through the AWGN channel with BPSK modulation scheme, using no coding and (5,7) convolutional hard- and soft-decision decoding using the Viterbi Algorithm. Simulation results are marked with ○ , □, or ▽. The Theoretical bound is the lower bound for transmission through the AWGN channel with BPSK modulation scheme.

A comparison of (5,7) convolutional coding using the Viterbi Decoder and (23,33) convolutional coding using the Viterbi Decoder, both with soft-decision, is shown in figure 5.3, and it is clear that the performance for (23,33) convolutional coding is approximately 0.4 dB better than the (5,7) convolutional coding at $BER = 10^{-3}$

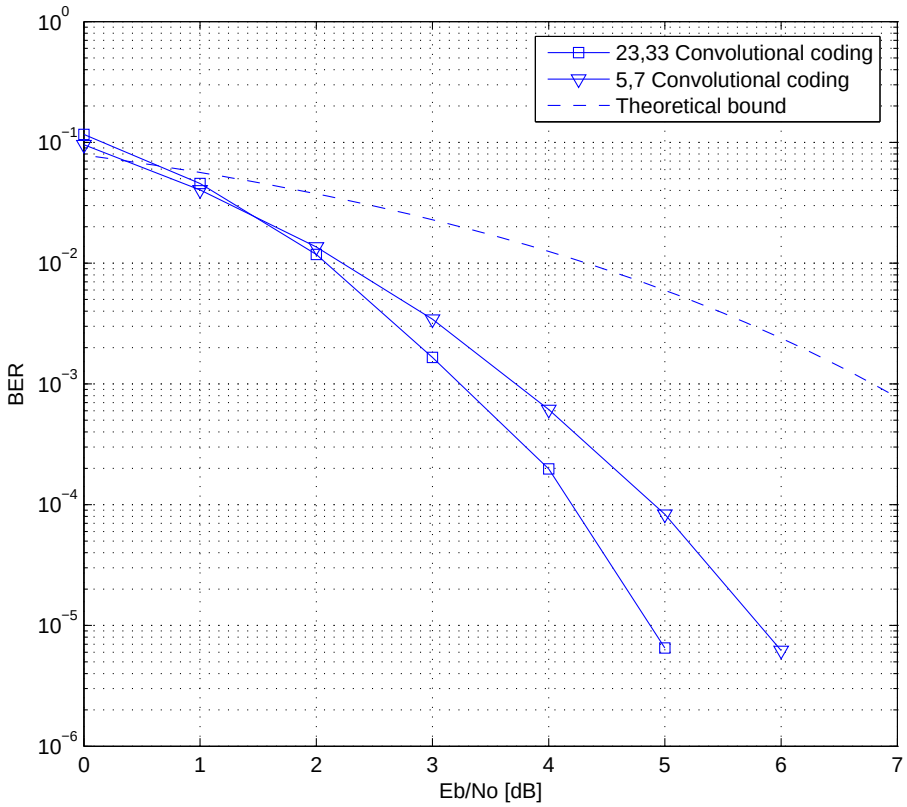


Figure 5.3: BER plot for transmission through the AWGN channel with BPSK modulation scheme, using (5,7) and (23,33) convolutional soft-decision decoding using the Viterbi Algorithm. Simulation results are marked with □, or ▽. The theoretical bound is the lower bound for transmission through the AWGN channel with BPSK modulation scheme.

CHAPTER 6

Equalization

InterSymbol Interference (ISI) is produced in the channel due to multipath propagation. A technique to eliminate (or reduce) the effect of ISI is to apply an *equalizer* after the matched filter. The equalizer should adapt to the channel characteristics, which are often evaluated runningly in real time applications such as e.g. the GSM system. In this project, we assume to have perfect knowledge of the channel characteristics at all time.

This chapter will describe two different equalizers, namely the Viterbi Equalizer and the Max-log-MAP Equalizer.

The input signal to the equalizer in the system model considered in this project is modelled by the channel encoder, the interleaver, the modulator, and the channel. The model for the input signal is described in the first section of this chapter.

The references used for this chapter are [Proakis et al., 2002] and [Morelos-Zaragoza, 2002], unless others are stated.

6.1 Signal Model

The input to the equalizer is called the *signal model* and is a result of the encoding process, the interleaving process, the modulation process, and transmission through the channel. A block diagram of these processes is shown in figure 6.1.

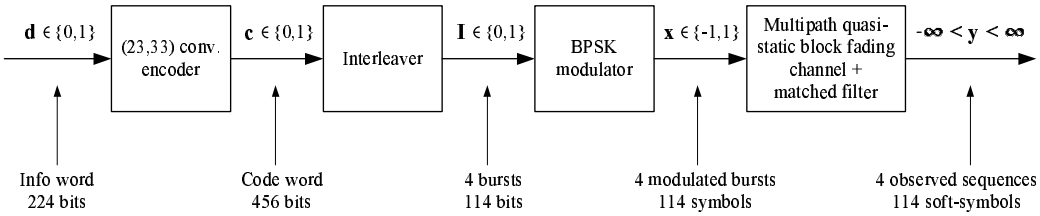


Figure 6.1: Block diagram of the signal model.

The input to the (23,33) convolutional encoder is a sequence, denoted as **d**, of 224 bits in $\{0, 1\}$. This sequence is called the info word, and the elements, denoted as d_i, are generated randomly.

The output sequence of the encoder is the input sequence to the interleaver and is called the code word. The code word is denoted as **c** and consists of 456 bits in $\{0, 1\}$. 4 tail bits are added to the 224 bits in the info word, and the new sequence of 228 bits are mapped into the code word as

$$c_i = \{c_i^{(1)} \;\; c_i^{(2)}\} = \{c_{2i-1} \;\; c_{2i}\} \qquad \text{for } i = 1, 2, \ldots, 228 \tag{6.1}$$

where

$$c_i^{(1)} = d_i \oplus d_{i-3} \oplus d_{i-4} \tag{6.2}$$
$$c_i^{(2)} = d_i \oplus d_{i-1} \oplus d_{i-3} \oplus d_{i-4} \tag{6.3}$$

$\oplus$ denotes modulo-2 addition.

Equation 6.1 can also be expressed as

$$c_k = \begin{cases} d_l \oplus d_{l-3} \oplus d_{l-4} & \text{for } k \text{ is odd} \\ d_m \oplus d_{m-1} \oplus d_{m-3} \oplus d_{m-4} & \text{for } k \text{ is even} \end{cases} \tag{6.4}$$

for $k = 1, 2, \ldots, 456$. $l = \frac{k+1}{2}$ and $m = \frac{k}{2}$.

The code word of 456 bits is the input to the interleaver. The code word is interleaved over 4 blocks (or sequences) and transmitted sequentially in 4 bursts. The interleaver sequence, consisting of 4 bursts, is denoted as **I** = $\{\mathbf{e}_1 \; \mathbf{e}_2 \; \mathbf{e}_3 \; \mathbf{e}_4\}$, and each element in $\mathbf{e}_B$, denoted as $e_{B,j}$ is computed as

$$e_{B,j} = c_k \qquad \text{for } k = 1, 2, \ldots, 456 \tag{6.5}$$

where

$$B = [(k-1) \bmod 4] + 1 \tag{6.6}$$
$$j = 2[49(k-1) \bmod 57] + [(k-1) \bmod 8] \operatorname{div} 4 \tag{6.7}$$

$B \in \{1, 2, 3, 4\}$ is the burst number, and $j \in \{1, 2, \ldots, 114\}$ is the bit number in burst B.

The input to the modulator is the interleaver sequence, and each element in the sequence is mapped as

$$x_k = \begin{cases} -1 & \text{for } I_k = 0 \\ +1 & \text{for } I_k = 1 \end{cases} \tag{6.8}$$

where x_k is an element in the transmitted sequence, denoted as **x**, and I_k is an element in **I**.

In chapter 2 we expressed the sampled output of matched filter as a vector

$$\underline{y} = \underline{\underline{R}}\,\underline{x} + \underline{\underline{A}}^H \underline{n} \tag{6.9}$$

where $\underline{\underline{A}}^H$ is the impulse response of the matched filter, $\underline{\underline{R}} = \underline{\underline{A}}^H \underline{\underline{A}}$ is the autocorrelation matrix of the channel, and $\underline{n}$ is the realization of a white Gaussian noise process.

$\underline{\underline{A}}^H$ is defined as

$$\underline{\underline{A}}^H = \begin{bmatrix} a_1^* & a_2^* & a_3^* & a_4^* & 0 & 0 & 0 & \dots \\ 0 & a_1^* & a_2^* & a_3^* & a_4^* & 0 & 0 & \ddots \\ 0 & 0 & a_1^* & a_2^* & a_3^* & a_4^* & 0 & \ddots \\ 0 & 0 & 0 & a_1^* & a_2^* & a_3^* & a_4^* & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix} \tag{6.10}$$

where a_1, a_2, a_3, and a_4 are the coefficients in the channel, $\underline{x}$ is the sampled input sequence to the channel represented as a vector, and $\underline{n}$ is the realization of $n(t)$. $a_l = z_n + jz_n$ for $l = 1, 2, 3, 4$, where z_n is a realization of a normalized random process with variance 1 and zero-mean. j denotes here the imaginary part.

We consider the output of the matched filter as a sequence of 456 bits, corresponding to one code word, and we denote it as **y**. Each element in **y** is computed as

$$y_k = \left(\underline{\underline{R}}\,\underline{x}\right)_k + \left(\underline{\underline{A}}^H \underline{n}\right)_k \tag{6.11}$$

The last term in equation 6.11 may be expressed as

$$\left(\underline{\underline{A}}^H \underline{n}\right)_k = a_1 n_k + a_2 n_{k-1} + a_3 n_{k-2} + a_4 n_{k-3} \tag{6.12}$$

where

$$n_k = \sqrt{\frac{1}{\frac{E_b}{N_0}}} \cdot z_n \tag{6.13}$$

z_n is a realization of a normalized random process with variance 1 and zero-mean. For further details about generating n_k we refer to appendix A.4.1 on page 71 concerning generation the additive white Gaussian noise.

The first term in equation 6.11 may be expressed as

$$\left(\underline{\underline{R}}\,\underline{x}\right)_k = \varrho_{41} x_{k+3} + \varrho_{31} x_{k+2} + \varrho_{21} x_{k+1} + \varrho_{11} x_k + \varrho_{12} x_{k-1} + \varrho_{13} x_{k-2} + \varrho_{14} x_{k-3} \tag{6.14}$$

where

$$\varrho_{11} = a_1^2 + a_2^2 + a_3^2 + a_4^2 \tag{6.15}$$

$$\varrho_{21} = \varrho_{12} = a_1 a_2 + a_2 a_3 + a_3 a_4 \tag{6.16}$$

$$\varrho_{31} = \varrho_{13} = a_1 a_3 + a_2 a_4 \tag{6.17}$$

$$\varrho_{41} = \varrho_{14} = a_1 a_4 \tag{6.18}$$

6.2 Maximum-Likelihood Sequence Detection

The equalizer finds the most likely sequence of transmitted symbols based on the observed symbols and knowledge of the channel. A model of the channel, the matched filter, and the equalizer is shown in figure 6.2, where $\underline{x}$ is the transmitted code word, $\underline{\underline{A}}$ is the channel, $\underline{n}$ is a realization of a white Gaussian noise process, $\underline{\underline{A}}^H$ is the matched filter, $\underline{y}$ is the observed code word, specified in equation 6.9, and $\underline{\widehat{x}}$ is the estimated code word.

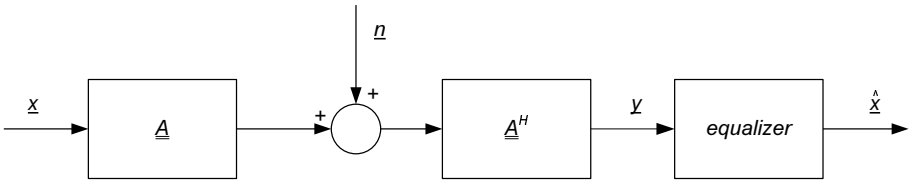


Figure 6.2: A model of the channel, matched filter and equalizer

The estimated sequence of code bits, denoted as $\underline{\widehat{x}}$, is the sequence with the highest probability of the observed sequence, $\underline{y}$

$$\begin{aligned}
\underline{\widehat{x}} &= \underset{\underline{x}\in\mathbb{X}}{\arg\max}\ P(\underline{y} \mid \underline{x}) && (6.19)\\
&= \underset{\underline{x}\in\mathbb{X}}{\arg\max}\left\{\frac{1}{\sqrt{2\pi\sigma_n}} e^{-\frac{(\underline{y}-\underline{x})^2}{2\sigma_n^2}}\right\} && (6.20)\\
&= \underset{\underline{x}\in\mathbb{X}}{\arg\max}\left\{\log\frac{1}{\sqrt{2\pi\sigma_n}}\left(-\frac{(\underline{y}-\underline{x})^2}{2\sigma_n^2}\right)\right\} && (6.21)\\
&= \underset{\underline{x}\in\mathbb{X}}{\arg\max}\left\{-(\underline{y}-\underline{x})^2\right\} && (6.22)
\end{aligned}$$

where $\mathbb{X}$ is the set of possible code words, and σ_n^2 is the noise variance.

The channel corrupts the transmitted signal by white Gaussian noise, but after the matched filter the noise is no longer white, which makes it more difficult to remove. To compensate for this, and make equation 6.22 usable, a whitening filter is applied to the observed signal described in equation 6.9

$$\begin{aligned}
(\underline{\underline{A}}^H)^{-1}\,\underline{y} &= (\underline{\underline{A}}^H)^{-1}\,\underline{\underline{R}}\,\underline{x} + (\underline{\underline{A}}^H)^{-1}\,\underline{\underline{A}}^H\,\underline{n} && (6.23)\\
\underline{z} &= \underline{\underline{A}}\,\underline{x} + \underline{n} && (6.24)
\end{aligned}$$

where $\underline{z} = (\underline{\underline{A}}^H)^{-1}\underline{y}$ is the output of the whitening filter.

The most likely sequence is when $\underline{n}$ is minimized, thus equation 6.22 can be written as equation 6.25, using the least square estimate.

$$\underline{\widehat{x}} = \underset{\underline{x}\in\mathbb{X}}{\arg\min}\left\{\|\underline{z} - \underline{\underline{A}}\,\underline{x}\|^2\right\} \qquad (6.25)$$

Equation 6.25 can be rewritten as

$$\widehat{\underline{x}} = \arg\min_{\underline{x}\in\mathbb{X}}\Big\{(\underline{z}-\underline{\underline{A}}\,\underline{x})^H(\underline{z}-\underline{\underline{A}}\,\underline{x})\Big\} \tag{6.26}$$

$$= \arg\min_{\underline{x}\in\mathbb{X}}\Big\{\underline{z}^H\,\underline{z}+\underline{x}^H\,\underline{\underline{A}}^H\,\underline{\underline{A}}\,\underline{x}-2Re\{\underline{x}^H\,\underline{\underline{A}}^H\,\underline{z}\}\Big\} \tag{6.27}$$

As we want to minimize with respect to $\underline{x}$, $\underline{z}^H\underline{z}$ can be omitted. Note that $\underline{\underline{A}}^H\underline{z}=\underline{y}$.

$$\widehat{\underline{x}} = \arg\min_{\underline{x}\in\mathbb{X}}\Big\{\underline{x}^H\,\underline{\underline{R}}\,\underline{x}-2Re\{\underline{x}^H\,\underline{\underline{A}}^H\,\underline{z}\}\Big\} \tag{6.28}$$

$$= \arg\min_{\underline{x}\in\mathbb{X}}\Big\{\sum_i\sum_j\varrho_{ij}\,x_i^*\,x_j-2Re\{\sum_i x_i^*\,y_i\}\Big\} \tag{6.29}$$

$$= \arg\min_{\underline{x}\in\mathbb{X}}\Big\{\sum_i|x_i|^2+2Re\{\sum_i\sum_{j=0}^{i-1}\varrho_{ij}\,x_i^*\,x_j\}-2Re\{\sum_i x_i^*\,y_i\}\Big\} \tag{6.30}$$

where ϱ_{ij} is the element in $\underline{\underline{R}}$ on row i, column j, and x_i is the i'th element in $\underline{x}$. The transition from equation 6.29 to equation 6.30 is due to the fact that $\varrho_{i,j}=\varrho_{j,i}^*$, which can easily be shown using equation 2.17 on page 19 and $\underline{\underline{R}}=\underline{\underline{A}}^H\underline{\underline{A}}$.

Since $x_i=\pm1\Rightarrow|x_i|^2=1$, $\sum_i|x_i|^2$ can be omitted.

$$\widehat{\underline{x}} = \arg\min_{\underline{x}\in\mathbb{X}}\Big\{2Re\{\sum_i\sum_{j=0}^{i-1}\varrho_{ij}\,x_i^*\,x_j\}-2Re\{\sum_i x_i^*\,y_i\}\Big\} \tag{6.31}$$

$$= \arg\min_{\underline{x}\in\mathbb{X}}\Big\{Re\{\sum_i(\sum_{j=0}^{i-1}(\varrho_{ij}\,x_i^*\,x_j)-x_i^*\,y_i\}\Big\} \tag{6.32}$$

$$= \arg\min_{\underline{x}\in\mathbb{X}}\Big\{Re\{\sum_i x_i^*(\sum_{j=0}^{i-1}(\varrho_{ij}\,x_j)-y_i)\}\Big\} \tag{6.33}$$

This means that we need to minimize the expression in equation 6.33, in the following called the metric. The branch metric, shown i equation 6.34, can be calculated for each possible branch.

$$Re\{x_i^*(\sum_{j=0}^{i-1}(\varrho_{ij}\,x_j)-y_i)\} \tag{6.34}$$

Notice that the whitening filter was just used for derivation and is not needed in the implementation of the equalizer.

6.3 Viterbi Equalizer

Maximum likelihood sequence detection is an optimum detection, and is often used when having a small number of states and/or short sequences. The basic idea is to compare the received sequence with all the possible sequences and then calculate the corresponding likelihoods. The sequence with the highest likelihood is then selected as the estimate of the transmitted sequence.

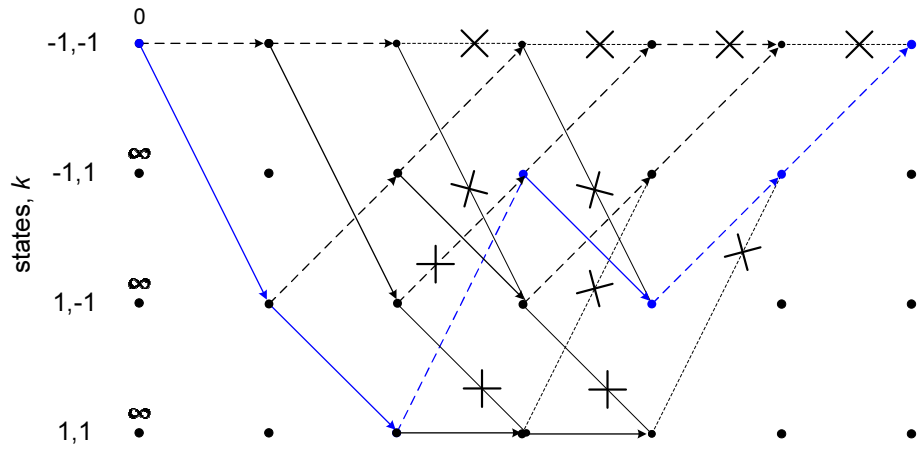


Figure 6.3: A trellis diagram used in the Viterbi Algorithm. The blue path is the most likely path. The crosses show which paths has been omitted.

The channel can be expressed by a trellis diagram, as shown in figure 6.3, where each path through the diagram corresponds to a different sequence.

The Viterbi Algorithm, described in chapter 5 on page 35, is a maximum likelihood detection method using the fact that only the most likely path to a given state at a given stage can be a part of the most likely path to another stage, thus all other paths to this first state can be discarded.
As stated in equation 6.33 the sum of the metric of each transition in the trellis must be minimized To find the most likely sequence it is necessary to go through the trellis diagram from left to right, and for each node find the most likely sequence (called the survivor) and discard the other. If each node has a value ξ which is the weight of the most likely path, this ξ can be calculated as

$$\xi_{i,k} = \max_j [\xi_{i,j} + \zeta_{i,j,k}] \tag{6.35}$$

where $\xi_{i,k}$ is the value of the node at stage i, state k, and $\zeta_{i,j,k}$ is the branch metric between state j and k at stage i.

A number of zeros, corresponding to the number of paths in the channel, has to be added to each sequence before modulation. This is to make sure the finishing state is known. It is then possible to backtrack from this state, through the survivors, thus finding the most likely path.

When the most likely sequence has been found, it is used as the output of the equalizer. This output is called hard output, as it only contains a sequence of 1's and 0's, and thus discards some information.

6.4 Max-log-MAP Equalizer

The Viterbi Equalizer described in the previous section defines the most likely sequence of symbols.

The output of the equalizer is often further processed. For instance a decoder is applied if the info word in encoded. To increase the amount of information passed to the decoder, and thereby possibly increasing the performance, an extended algorithm is needed. This algorithm must result

in not only the estimated sequence, but also an estimate of the reliability of each symbol in the sequence. The Max-log-MAP algorithm, described in this section, has this property, and it results in a soft value representation for each symbol, namely the logarithm of the *Maximum A posteriori Probability* (MAP).

In this section the MAP algorithm is described first, and then the modification, Max-log-MAP, is shown.

6.4.1 MAP

We define the probabilities α, β and γ in the following way [Bahl et al., 1974]

$$\alpha_i^{(k)} \triangleq P(x_i = u_k \ ; \ y_1, y_2, ..., y_i) \tag{6.36}$$

$$\beta_i^{(k)} \triangleq P(y_{i+1}, y_{i+2}, ..., y_L \mid x_i = u_k) \tag{6.37}$$

$$\gamma_i^{(j,k)} \triangleq P(x_i = u_k \ ; \ y_i \mid x_{i-1} = u_j) \tag{6.38}$$

where $\mathbf{x}= [x_1, x_2, ..., x_L]$ is the transmitted sequence of symbols, $\mathbf{y}= [y_1, y_2, ..., y_L]$ is the received sequence, $u_k \in \{-1, +1\}$ is the value of x_i corresponding to state k, and L is the length of the sequence.

The MAP algorithm also uses the trellis diagram. Each node in the trellis has an α and a β attached, and each transition has a γ attached. Since there is several transitions to each node, equations 6.36 and 6.37 can be rewritten as

$$\alpha_i^{(k)} = \sum_j P(x_{i-1} = u_j \ , \ y_1, y_2, ..., y_{i-1}) \cdot P(x_i = u_k \ , \ y_i \mid x_{i-1} = u_j) \tag{6.39}$$

$$= \sum_j \alpha_{i-1}^{(j)} \cdot \gamma_i^{(j,k)} \tag{6.40}$$

$$\beta_i^{(k)} = \sum_j P(y_{i+2}, y_{i+3}, ..., y_L \mid x_{i+1} = u_j) \cdot P(x_{i+1} = u_j \ , \ y_{i+1} \mid x_i = u_k) \tag{6.41}$$

$$= \sum_j \beta_{i+1}^{(j)} \cdot \gamma_{i+1}^{(k,j)} \tag{6.42}$$

for $j = 1, 2, \ldots, M$, where M is the number of states.

The a posteriori probability of state k at stage i is defined as

$$P(x_i = u_k|\mathbf{y}) = \sum_j P(x_{i-1} = u_j \ , \ x_i = u_k \ , \ y_1, y_2, ..., y_L) \tag{6.43}$$

$$= \sum_j P(x_{i-1} = u_j \ , \ y_1, y_2, ..., y_L) \cdot P(y_{i+1}, ..., y_L \mid x_i = u_k) \tag{6.44}$$
$$\cdot P(x_i = u_k \ , \ y_i \mid x_{i-1} = u_j)$$

$$= \sum_j \alpha_{i-1}^{(j)} \cdot \beta_i^{(k)} \cdot \gamma_i^{(j,k)} \tag{6.45}$$

6.4.2 Log-MAP

Multiplication is rather computational expensive, thus an approach to decrease the complexity, the log-MAP Algorithm is introduced. The log-probabilities α', β' and γ' are then computed as

$$\alpha_i'^{(k)} = \log\ \alpha_i^{(k)} = \log\Big(\sum_j \alpha_{i-1}^{(j)} \cdot \gamma_i^{(j,k)}\Big) \tag{6.46}$$

$$\beta_i'^{(k)} = \log\ \beta_i^{(k)} = \log\Big(\sum_j \beta_{i+1}^{(j)} \cdot \gamma_{i+1}^{(k,j)}\Big) \tag{6.47}$$

$$\gamma_i'^{(j,k)} = \log\ \gamma_i^{(j,k)} \tag{6.48}$$

The branch metric defined in equation 6.34 can also be used for log-MAP since it is based on the logarithm of the probabilities.

$$\gamma_i'^{(j,k)} = Re\{x_i^*(\sum_{j=0}^{i-1}(\varrho_{ij}\ x_j) - y_i)\} \tag{6.49}$$

where ϱ is the elements of $\underline{\underline{R}}$, x_i is an element in **x**, and y_i is an element in **y**.

6.4.3 Max-log-MAP

The α''s and β''s in equations 6.46 and 6.47 are still computational expensive, so the assumption in equation 6.51 is used to reduce complexity, resulting in equation 6.54 and 6.57.

$$\log(\delta_1 + \delta_2) = \max\big[\log(\delta_1), \log(\delta_2)\big] + \log\big(1 + e^{-|\log\delta_1 + \log\delta_2|}\big) \tag{6.50}$$

$$\approx \max\big[\log(\delta_1), \log(\delta_2)\big] \tag{6.51}$$

$$\alpha_i'^{(k)} \approx \max_j\Big[\log\Big(\alpha_{i-1}^{(j)} \cdot \gamma_i^{(j,k)}\Big)\Big] \tag{6.52}$$

$$= \max_j\Big[\log\Big(\alpha_{i-1}^{(j)}\Big) + \log\Big(\gamma_i^{(j,k)}\Big)\Big] \tag{6.53}$$

$$= \max_j\Big[\alpha_{i-1}'^{(j)} + \gamma_i'^{(j,k)}\Big] \tag{6.54}$$

$$\beta_i'^{(k)} \approx \max_j\Big[\log\Big(\beta_{i+1}^{(j)} \cdot \gamma_{i+1}^{(k,j)}\Big)\Big] \tag{6.55}$$

$$= \max_j\Big[\log\Big(\beta_{i+1}^{(j)}\Big) + \log\Big(\gamma_{i+1}^{(k,j)}\Big)\Big] \tag{6.56}$$

$$= \max_j\Big[\beta_{i+1}'^{(j)} + \gamma_{i+1}'^{(k,j)}\Big] \tag{6.57}$$

This can be represented in a trellis diagram like for the Viterbi Algorithm, but the difference is that each node in the trellis diagram has now two values, $\alpha_i'^{(k)}$ and $\beta_i'^{(k)}$ respectively, and the branches have the values $\gamma_i'^{(k,j)}$. Such a trellis diagram for a channel with a memory depth of 1 is shown in figure 6.4.

Based on the γ''s, the α''s can now be computed going through the trellis diagram from left to right, while the β''s can be computed going the other way, from right to left. Note that $\alpha_0'^{(-1)} = \beta_L'^{(-1)} = 0$ and $\alpha_0'^{(+1)} = \beta_L'^{(+1)} = \infty$.

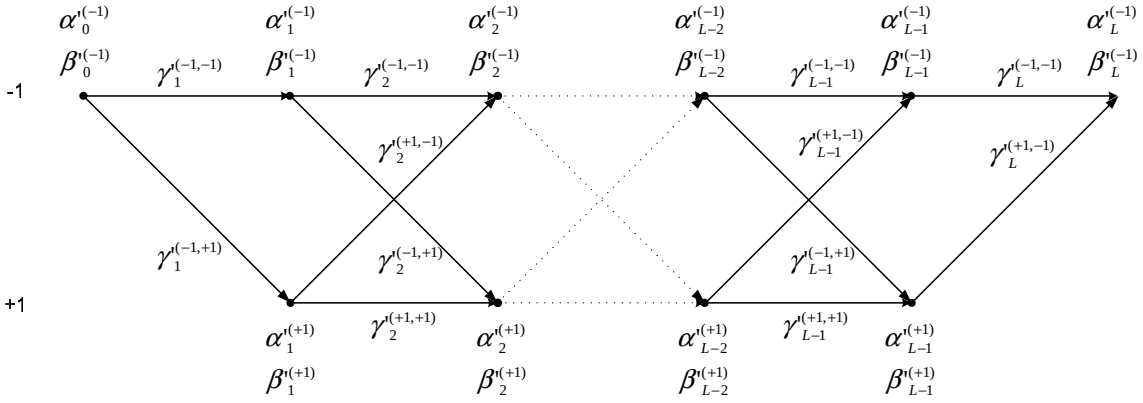


Figure 6.4: A trellis diagram used in Max-log-MAP equalization.

When both the α''s and the β''s are known, we can, based on equation 6.45 on page 46, use the following equation to calculate the Max-log-MAP of each node, denoted as $\delta_k(i)$, in the trellis diagram.

$$\begin{aligned} \delta_i^{(k)} &= \alpha_i^{(k)} \cdot \beta_i^{(k)} && (6.58) \\ &= \alpha_i'^{(k)} + \beta_i'^{(k)} && (6.59) \end{aligned}$$

The output of the Max-log-MAP Equalizer for the i'th symbol, denoted as Δ_i, is the log-likelihood ratio shown in equation 6.60

$$\begin{aligned} \Delta_i &= \log \frac{P(x_i = +1|\mathbf{y})}{P(x_i = -1|\mathbf{y})} && (6.60) \\ &= \log P(x_i = +1|\mathbf{y}) - \log P(x_i = -1|\mathbf{y}) && (6.61) \\ &\approx \max_{k|_{u_k=+1}} \delta_i^{(k)} - \max_{k|_{u_k=-1}} \delta_i^{(k)} && (6.62) \end{aligned}$$

Note that hard-decision on the output of the Max-log-MAP Algorithm gives the same sequence as the output of the Viterbi Algorithm.

6.5 Performance

To test the performance of the equalizer, a large number of bursts consisting of 224 bits, have been mapped into BPSK symbols and transmitted through a channel modelled as the 4-path quasi-static block fading channel specified in chapter 1, and a matched filter.

The observed sequences are then equalized. The resulting BER for different $\frac{E_b}{N_0}$ is shown in figure 6.5

It is clear, that the equalizer improves the performance.

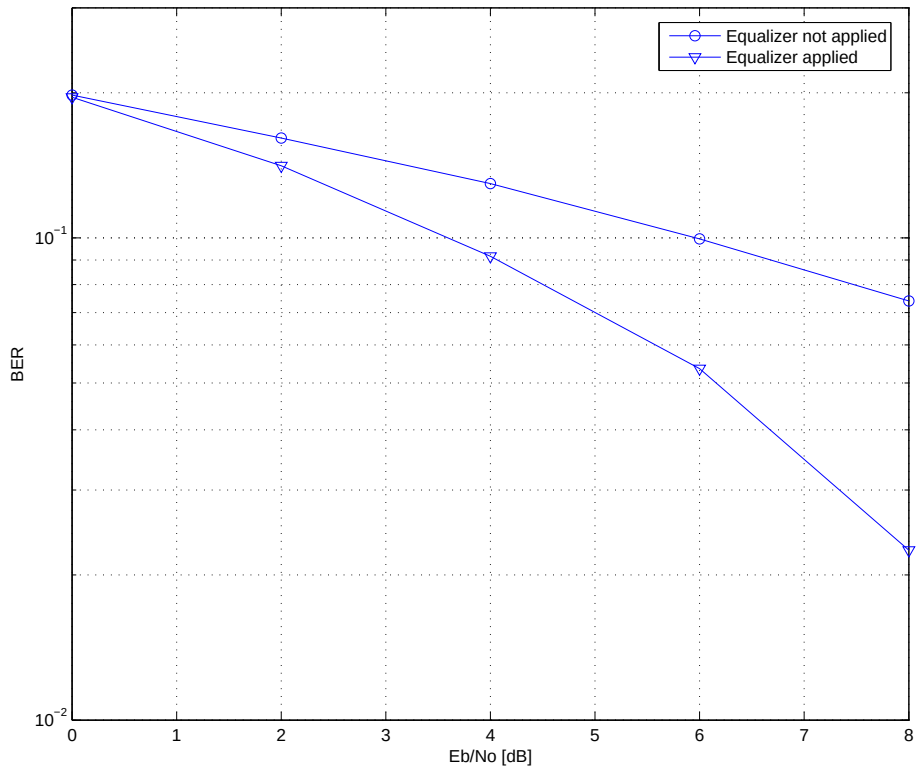


Figure 6.5: Performance of the equalizer. The channel is the 4-path quasi-static block fading channel.

CHAPTER 7

Iterative Equalization and Decoding

The GSM standard does not specify any receiver, but a commonly used receiver structure is sequential equalization and decoding. In this project, we seek to design a receiver that will improve the performance.

A coding scheme known to approach the Shannon limit is turbo-coding. It is introduced in [Berrou et al., 1993] and uses an iterative technique. The iterative technique can be employed to the equalizer, deinterleaver and decoder in the receiver of the GSM system.

In this chapter the simple, non-iterative receiver is described. Next turbo-decoding is described, and finally the iterative receiver considering both equalization and decoding is described.

The primary reference for this chapter is [Proakis et al., 2002], unless others are stated.

7.1 The Non-Iterative Receiver

A block diagram of the non-iterative receiver, incorporating the equalizer, deinterleaver, and decoder is shown in figure 7.1, where $\mathbf{y}$ is the observed sequence from the matched filter, and $\widehat{\mathbf{d}}$ is the estimated info word.

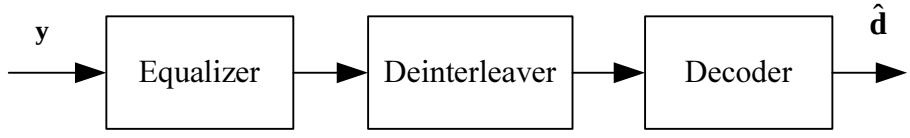


Figure 7.1: A non-iterative receiver.

To test the performance, a large number of info words of 224 bits have been encoded using (23,33) convolutional coding, interleaved and divided into 4 bursts, each consisting of 114 bits.

These bursts are each mapped into BPSK symbols and transmitted through the 4-path quasi-static block fading channel specified in chapter 1, and a matched filter.

The observed sequences are then equalized using a Max-log-MAP equalizer and decoded using a soft-input Viterbi Decoder.

The resulting BER for different $\frac{E_b}{N_0}$ is shown in figure 7.2, compared to the performance using the same transmitting and receiving scheme, but without encoding and decoding.

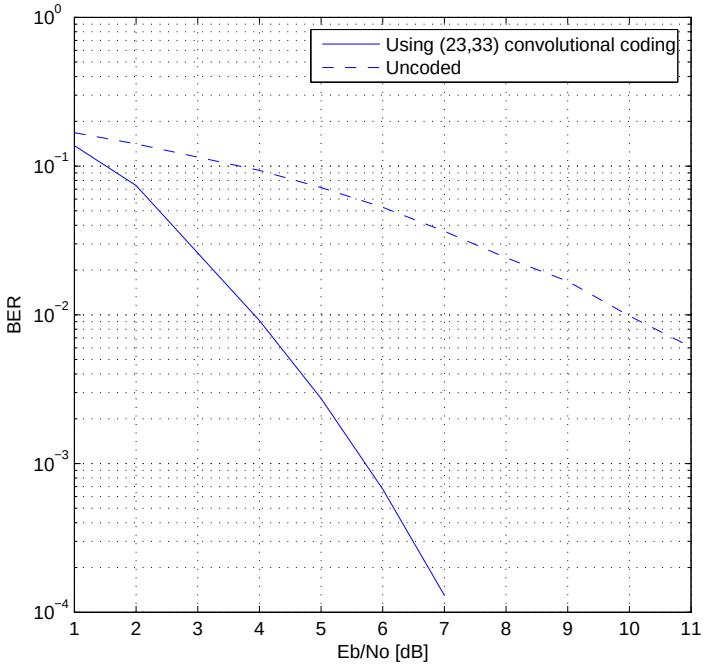


Figure 7.2: Performance of the non-iterative receiver.

7.2 Turbo Decoding

Iterative decoding was first introduced in [Berrou et al., 1993], in a method called Turbo decoding. By using the iterative technique, the performance increases for each new iteration. This result is a motivation to implement the iterative technique in the GSM receiver.

7.2.1 Parallel Concatenated Codes

The turbo coding/decoding method described in [Berrou et al., 1993] considers a parallel concatenated coding scheme consisting of two encoders. The encoding scheme is shown in figure 7.3.

The transmitted info word $\mathbf{d}$ is represented and transmitted through the channel as $\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3$ in the following way. $\mathbf{c}_1$ is uncoded, $\mathbf{c}_2$ is the encoded $\mathbf{d}$ using Encoder 1 and $\mathbf{c}_3$ is the interleaved and encoded $\mathbf{d}$ using Encoder 2.

Both encoders use the same input bits, $\mathbf{d}$, but ordered differently due to the presence of the interleaver. This minimizes the correlation between the two sequences of code symbols, $\mathbf{c}_2$ and $\mathbf{c}_3$, thus improving the probability of recovering the encoded bits.

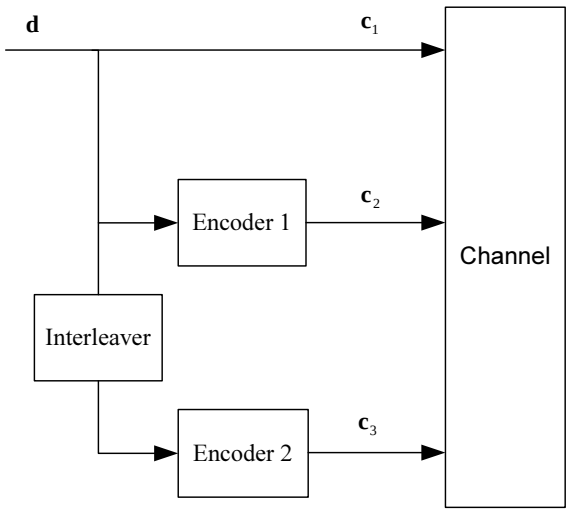


Figure 7.3: Block diagram of the turbo-encoding scheme.

7.2.2 Iterative Decoding

The iterative decoder scheme of turbo-decoding is shown in figure 7.4

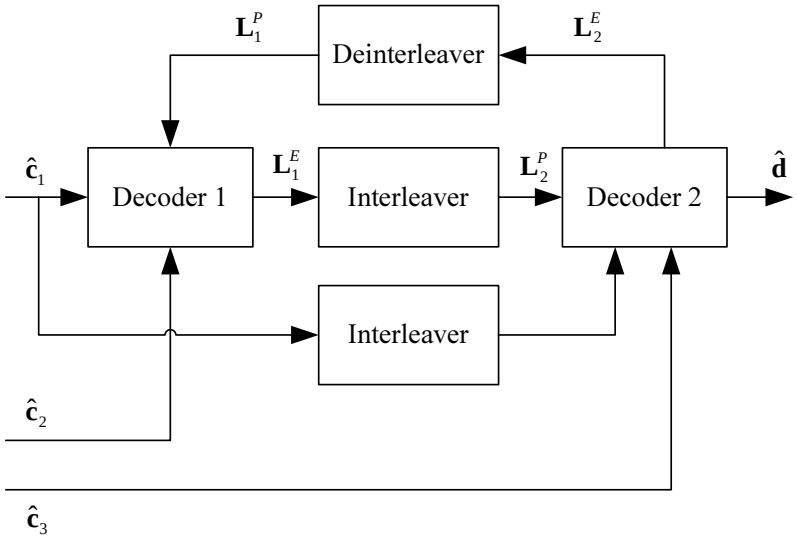


Figure 7.4: Iterative decoding scheme of turbo-codes.

The iterative decoding scheme consists of the following steps:

1. First the received sequences, $\widehat{\mathbf{c}}_1$ and $\widehat{\mathbf{c}}_2$, are decoded in Decoder 1, and the resulting sequence of extrinsic information, denoted as $\mathbf{L}_1^E$, is interleaved and passed to Decoder 2 as $\mathbf{L}_2^P$.

2. The extrinsic information, $\mathbf{L}_2^E$, resulting from decoding of both $\mathbf{L}_2^P$, $\widehat{\mathbf{c}}_3$ and the interleaved $\widehat{\mathbf{c}}_1$ in Decoder 2, is deinterleaved and passed to Decoder 1 as $\mathbf{L}_1^P$.

3. The new extrinsic information, $\mathbf{L}_1^E$, resulting from decoding of both $\mathbf{L}_1^P$, $\widehat{\mathbf{c}}_1$ and $\widehat{\mathbf{c}}_2$ in Decoder 1, is interleaved and passed to Decoder 2.

4. Step 2 and 3 are repeated for each following iteration.

5. An estimation, $\widehat{\mathbf{d}}$, of the transmitted bit-sequence, $\mathbf{d}$, is finally found based on the output of Decoder 2.

Note that decoding delays must be disregarded, so that $\widehat{\mathbf{x}}_1$, $\widehat{\mathbf{x}}_2$ and $\widehat{\mathbf{x}}_3$ are kept in a buffer throughout the decoding process.

7.2.3 Extrinsic Information

A vital part of turbo decoding is the *extrinsic information*. It is described in the following. Consider the output sequence of a Max-log-MAP algorithm, denoted as $\mathbf{\Delta}$. It is in chapter 6 defined as

$$\begin{aligned}
\Delta_i &= \log \frac{P(d_i = 1|\mathbf{c})}{P(d_i = 0|\mathbf{c})} && (7.1)\\
&= \log \frac{P(\mathbf{c}|d_i = 1)}{P(\mathbf{c}|d_i = 0)} \cdot \frac{P(d_i = 1)}{P(d_i = 0)} && (7.2)\\
&= \log \frac{P(\mathbf{c}_{\overline{i}}|d_i = 1)}{P(\mathbf{c}_{\overline{i}}|d_i = 0)} \cdot \frac{P(\mathbf{c}_i|d_i = 1)}{P(\mathbf{c}_i|d_i = 0)} \cdot \frac{P(d_i = 1)}{P(d_i = 0)} && (7.3)\\
&= \log \frac{P(d_i = 1|c_{\overline{i}})}{P(d_i = 0|c_{\overline{i}})} \cdot \frac{P(c_i|d_i = 1)}{P(c_i|d_i = 0)} && (7.4)\\
&= \log \frac{P(c_i|d_i = 1)}{P(c_i|d_i = 0)} + \log \frac{P(d_i = 1|c_{\overline{i}})}{P(d_i = 0|c_{\overline{i}})} && (7.5)
\end{aligned}$$

where $\mathbf{d} = \{d_1, d_2, ..., d_i, ..., d_L\}$ is the transmitted info word, $\mathbf{c} = \{c_1, c_2, ..., c_i, ..., c_L\}$ is the received code word, $\mathbf{c}_{\overline{i}}$ is $\mathbf{c}$ without c_i, and $\mathbf{\Delta} = \{\Delta_1, \Delta_2, ..., \Delta_i, ..., \Delta_L\}$ is the output of the decoder.

The two parts of equation 7.5 is called the *intrinsic information*, denoted as $\mathbf{L}^I_{c_i}$, and the *extrinsic information*, denoted as $\mathbf{L}^E_{c_i}$, respectively. The intrinsic information of each bit depends solely on the corresponding symbol of the input to the decoder. The extrinsic information is added by the decoding, and as such only depends on the redundant information introduced by the encoder.

Due to the interleaver, the sequences of extrinsic information passed between the decoders in the turbo-decoding scheme are weakly correlated with the observed sequences. They can therefore be used jointly in a new decoding process. [Berrou et al., 1993]

7.3 Iterative Equalization and Decoding

The iterative technique is in [Jordan et al., 1998] and [Bauch et al., 1998] applied to a sequential equalization and decoding scheme. The method is described in the following.

A non-iterative transmitter and receiver scheme is shown in figure 7.5. The channel used in this project incorporates ISI, and can therefore be regarded as a convolutional code with complex-valued code symbols. The encoder, interleaver and channel then form a serially concatenated encoding scheme.

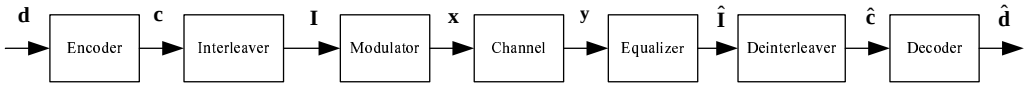


Figure 7.5: A non-iterative transmitter and receiver scheme.

It is shown in [Benedetto et al., 1998], that iterative decoding of serial concatenated codes, using extrinsic information, yields performance gains similar to those of turbo-decoding. Based on this, a iterative scheme using two serially concatenated elements is reasonable.

The modified receiver using the iterative scheme is shown in figure 7.6, where $\mathbf{y}$ is the observed sequence of code symbols from the matched filter, $\mathbf{L}_I$ is the sequence of log-likelihood ratios resulting from the equalizer, $\mathbf{L}_I^P$ is the interleaved sequence of extrinsic information resulting from the decoder, $\mathbf{L}_I^E$ is the sequence of extrinsic information resulting from the equalizer, $\mathbf{L}_c$ is the sequence of information resulting from the decoder, $\mathbf{L}_c^P$ is the deinterleaved sequence of extrinsic information resulting from the equalizer, $\mathbf{L}_c^E$ is the sequence of extrinsic information resulting from the decoder, and $\widehat{\mathbf{d}}$ is the estimated info word.

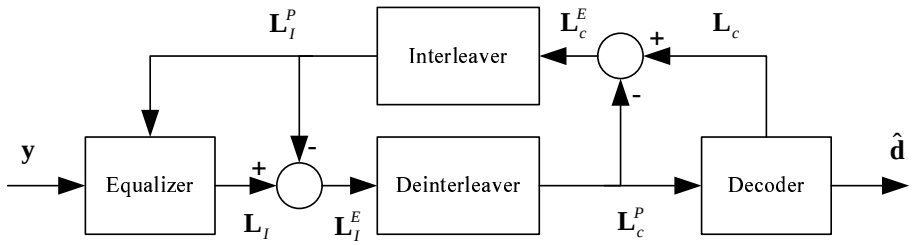


Figure 7.6: Receiver using iterative equalization and decoding.

This iterative receiver consists of the following steps.

1. First the observed sequence, $\mathbf{y}$, is equalized, and the resulting sequence of extrinsic information, $\mathbf{L}_I^E$, is deinterleaved and passed to the decoder.
2. The deinterleaved sequence, $\mathbf{L}_c^P$, is subtracted from the log-likelihood ratios, $\mathbf{L}_c$, resulting from decoding of $\mathbf{L}_c^P$. It is then interleaved and passed to the equalizer.
3. The interleaved sequence, $\mathbf{L}_I^P$, is subtracted from the new log-likelihood ratios, $\mathbf{L}_I$, resulting from equalization of $\mathbf{L}_I^P$ and $\mathbf{y}$. It is then deinterleaved and passed to the decoder.
4. Step 2 and 3 are repeated for each following iteration.
5. An estimation, $\widehat{\mathbf{d}}$, of the transmitted bit-sequence, $\mathbf{d}$, is finally found based on the output of the decoder.

Note that **y** is kept in a buffer throughout the iterative equalization and decoding.

7.3.1 Exchanging Information between Equalizer and Decoder

The subtractions in steps 2 and 3 on this page are necessary to find the extrinsic information, since outputs of both the equalizer, $\mathbf{L}_I$, and the decoder, $\mathbf{L}_c$, are *a posteriori informations* consisting of

both intrinsic and extrinsic information. The subtractions are due to

$$\begin{aligned} \mathbf{L}_{\mathbf{I}}^{E} &= \mathbf{L}_{\mathbf{I}} - \mathbf{L}_{\mathbf{I}}^{I} && (7.6)\\ &= \mathbf{L}_{\mathbf{I}} - \mathbf{L}_{\mathbf{I}}^{P} && (7.7)\\ \mathbf{L}_{\mathbf{c}}^{E} &= \mathbf{L}_{\mathbf{c}} - \mathbf{L}_{\mathbf{c}}^{I} && (7.8)\\ &= \mathbf{L}_{\mathbf{c}} - \mathbf{L}_{\mathbf{c}}^{P} && (7.9) \end{aligned}$$

where $\mathbf{L}_{\mathbf{I}}^{I}$ is the sequence of intrinsic information of the burst, and $\mathbf{L}_{\mathbf{c}}^{I}$ is the sequence of intrinsic information of the code word.

The equalizer in the modified scheme receives two different inputs. The observed sequence from the channel, $\mathbf{y}$, and the interleaved extrinsic information from the decoder, $\mathbf{L}_{\mathbf{I}}^{P}$.

The observed sequence, $\mathbf{y}$, is, as described in chapter 6 regarding equalization, used in computing the branch metrics, which are the log-probability of each branch in the trellis.

The interleaved extrinsic information, $\mathbf{L}_{\mathbf{I}}^{P}$, passed from the decoder, is a log-likelihood ratio of each symbol. It can be changed into a log-probability of each branch using the following equations.

$$\begin{aligned} \log P(I_i = 0) &= -\log\left(1 + e^{\log \frac{P(I_i=1)}{P(I_i=0)}}\right) && (7.10)\\ \log P(I_i = 1) &= \log \frac{P(I_i = 1)}{P(I_i = 0)} + \log P(I_i = 0) && (7.11) \end{aligned}$$

As the two probabilities are only weakly correlated due to the interleaver and the use of extrinsic information, they are regarded independent. Therefore the log-probabilities can be added (the probabilities can be multiplied), and this sum is used in the equalizer.

To preserve information in the iterative loop for each symbol, the Viterbi decoder described in chapter 5, must be modified into a Max-log-MAP decoder like the equalizer was modified in chapter 6. Thereby the extrinsic information can be extracted.

The output of a Max-log-MAP decoder is the log-likelihood ratio of the paths corresponding to each *info bit* being 1 compared to the paths corresponding to the info bit being 0. The decoder used in the implemented iterative scheme must be modified, so the output is the log-likelihood ratio of the paths corresponding to each *code bit* being 1 compared to the paths corresponding to the code bit being 0. As the used encoding has a code rate of 1/2, this means the output of the decoder must consist of 456 log-likelihood ratios, one for each code bit, where it previous was 224, one for each info bit (disregarding tail bits). This modification is straightforward, and will not be shown here.

7.4 Simulations and Results

Three performance tests have been conducted:

1. A 4-path quasi-static block Rayleigh fading channel is used as the channel model. The transmitter consists of a (23,33) convolutional encoder, an interleaver, and a BPSK modulator. The receiver consists of an iterative equalizer and decoder.
   This is a test of our considered system model specified in chapter 1.

2. A 4-path fixed channel is used as the channel model. The transmitter and the receiver is the same as for test 1.
   This test is an experiment. We wish to investigate the performance if no irrecoverable bursts occur due to fading in the channel, and if it is possible to achieve a performance gain with more iterations. The motivation for this experiment is to test the performance the iterative equalizer and decoder with another channel model scenario.

3. A 4-path quasi-static block Rayleigh fading channel is used as the channel model. The transmitter and the receiver is the same as for test 1, except that the interleaver size is increased.
   This test is another experiment. We wish to investigate the effect of the interleaver size.

7.4.1 Setup for simulations

We wish to investigate the performance by inspecting plots with *Bit Error Rate* (BER) vs. E_b/N_0. We are using Monte Carlo simulations, and for a specified E_b/N_0, sequences of 224 bits are transmitted through the system until either a predefined number of errors has occurred or a predefined maximum number of sequences has been transmitted. The bit error rate is calculated as

$$BER = \frac{\#\text{ bit errors}}{\#\text{ transmitted bits}} \tag{7.12}$$

where the number of bit errors is found by comparing the input bits to the system with the output bits.

The considered (23,33) convolutional encoder converts a info word of 224 info bits and 4 tail bits into a code word of 456 bits. The interleaver is the modified GSM interleaver specified in chapter 1, and it interleaves the code word of 456 bits over four sequentially transmitted bursts of 114 bits. The BPSK modulator maps the bits in the bursts of $\{0, 1\}$ into symbols of $\{-1, +1\}$.

The symbols from the modulator are transmitted through the specified channel (specified for each test). Two channels are considered: 1) The 4-path quasi-static block Rayleigh fading channel, where the coefficients are Rayleigh distributed and time-invariant during one burst of 114 bits, and for each new burst the coefficients change. 2) The 4-path fixed channel, where the coefficients are fixed to $[\sqrt{0.45}, \sqrt{0.25}, \sqrt{0.15}, \sqrt{0.15}]$.

After transmission through the channel and a matched filter, the four bursts (corresponding to one code word of 456 bits) are buffered. The equalizer, the deinterleaver, and the decoder is implemented in an iterative structure, where the equalizer and the decoder are exchanging extrinsic information with each other. Both the equalizer and the decoder are based on the Max-log-MAP Algorithm.

7.4.2 Tests

Test 1: 4-Path Quasi-Static Block Rayleigh Fading Channel

This test will consider the 4-path quasi-static block Rayleigh fading channel. A large number of info words have been transmitted, and each burst has been transmitted through an individual realization of the channel. The results from the simulation using iterative equalization and decoding is shown in figure 7.7.

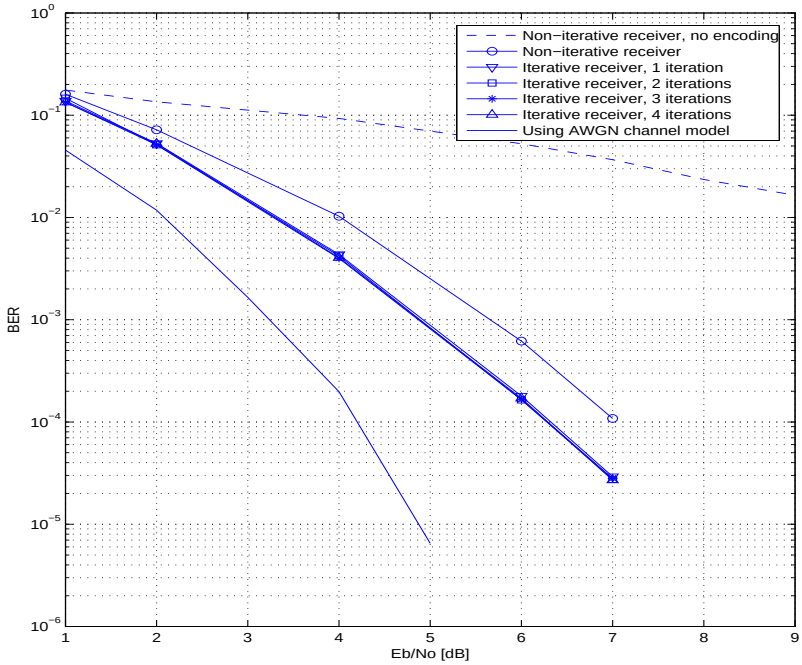


Figure 7.7: Performance of the iterative equalization and decoding with transmission through a 4-path quasi-static block fading channel.

In the figure we have plotted two curves for comparison with our results. One is the performance for transmission through the AWGN channel using (23,33) convolutional coding, and it is also the theoretical lower bound for our results. The other is the performance for transmission through the 4-path quasi-static block Rayleigh fading channel and with no coding.

Simulations has shown that there is a performance gain from the 0'th iteration (which is the non-iterative receiver scheme) to the 1'st iteration of 0.7 dB from E_b/N_0 = 5.6 dB to 4.9 dB at $BER = 10^{-3}$. The following three iterations (2'nd, 3'rd, and 4'th iteration) do not give any appreciable performance gain. The results improve only very little compared to the 1'st iteration. At $BER = 10^{-2}$, E_b/N_0 is improved from 4.0 dB to 3.3 dB, and at $BER = 10^{-4}$, E_b/N_0 is improved from 7.0 dB to 6.3 dB.

In [Jordan et al., 1998] a system model comparable to the one we have implemented, is evaluated. The differences are that [Jordan et al., 1998] have used a "pseudo random" interleaver, a scaling factor to damp the *a priori* information to the equalizer, and two different channel models, namely *Typical Urban* and *Hilly Terrain* specified in [Hoeher, 1992]. The results from their simulations for both channel models show a performance gain of approximately 0.7 dB for $BER = 10^{-3}$ from the non-iterative receiver scheme to the 1'st iteration. Furthermore, they do not get any performance improvement for the following iterations.

Test 2: 4-Path Fixed Channel

This test will consider a no-fading channel with the fixed coefficients [$\sqrt{0.45}$, $\sqrt{0.25}$, $\sqrt{0.15}$, $\sqrt{0.15}$]. A large number of info words have been transmitted, and the results from the simulation is shown in figure 7.8.

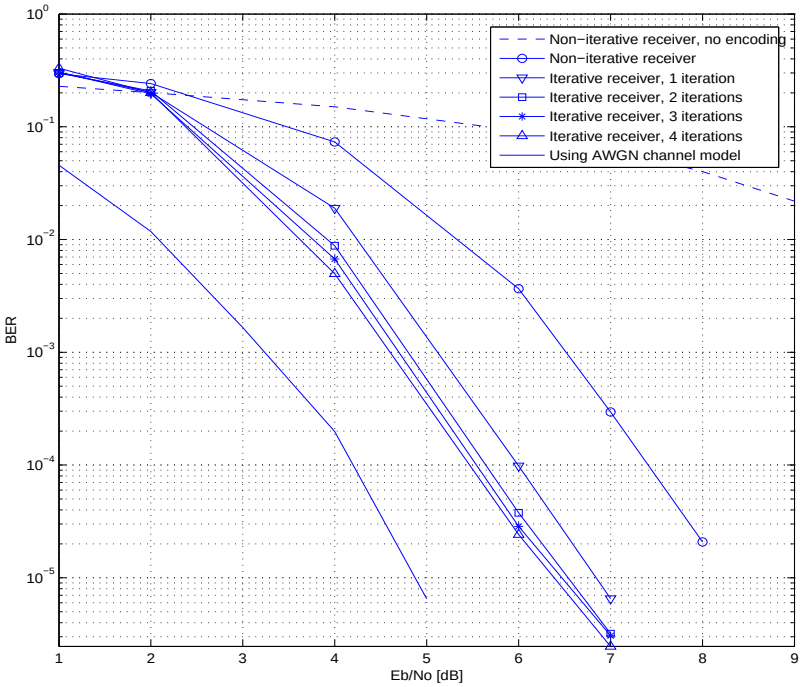


Figure 7.8: Performance of the iterative equalization and decoding with transmission through a 4-path fixed channel with the coefficients [$\sqrt{0.45}$, $\sqrt{0.25}$, $\sqrt{0.15}$, $\sqrt{0.15}$].

We have in this figure plotted the lower bound performance curve and the performance for transmission through the 4-path fixed channel using no coding for comparison with our results.

The simulations has shown a performance gain for each new iteration from the 0'th iteration to the 4'th iteration. E_B/N_0 is 6.5 dB for the 0'th iteration and 4.5 dB for the 4'th iteration at $BER = 10^{-3}$. The same performance gain of approximately 2 dB is also achieved at $BER = 10^{-2}$ and $BER = 10^{-4}$.

In [Jordan et al., 1998] a fixed coefficients channel is also used in their simulations. They use the five coefficients [$\sqrt{0.45}$, $\sqrt{0.25}$, $\sqrt{0.15}$, $\sqrt{0.10}$, $\sqrt{0.05}$], where we use only four coefficients, [$\sqrt{0.45}$, $\sqrt{0.25}$, $\sqrt{0.15}$, $\sqrt{0.15}$]. They achieve a performance gain from the 0'th iteration to the fourth iteration is approximately 2 dB at BER$= 10^{-3}$ with a decreasing performance gain for each new iteration.

Test 3: 4-Path Quasi-Static Block Rayleigh Fading Channel and Increased Interleaver Size

This test will consider the 4-path quasi-static block Rayleigh fading channel and an increased interleaver size. The simulation setup (and the GSM standard) specify an interleaver size of 456 bits, but in this test we use a size of 4104 bits.

The modified interleaver, which is used in the other tests, is defined by equation 6.5, 6.6, and 6.7 on page 41. With the increased size interleaver, the burst size is increased from 114 bits to 1026 bits, and the only change in the definition of the interleaver block is in equation 6.7, where

$$j = 2[49(k-1) \bmod 57] + [(k-1) \bmod 8] \operatorname{div} 4 \tag{7.13}$$

is changed to

$$j = 2[49(k-1) \bmod 513] + [(k-1) \bmod 8] \operatorname{div} 4 \tag{7.14}$$

A large number of info words have been transmitted, and the results from the simulation is shown in figure 7.9.

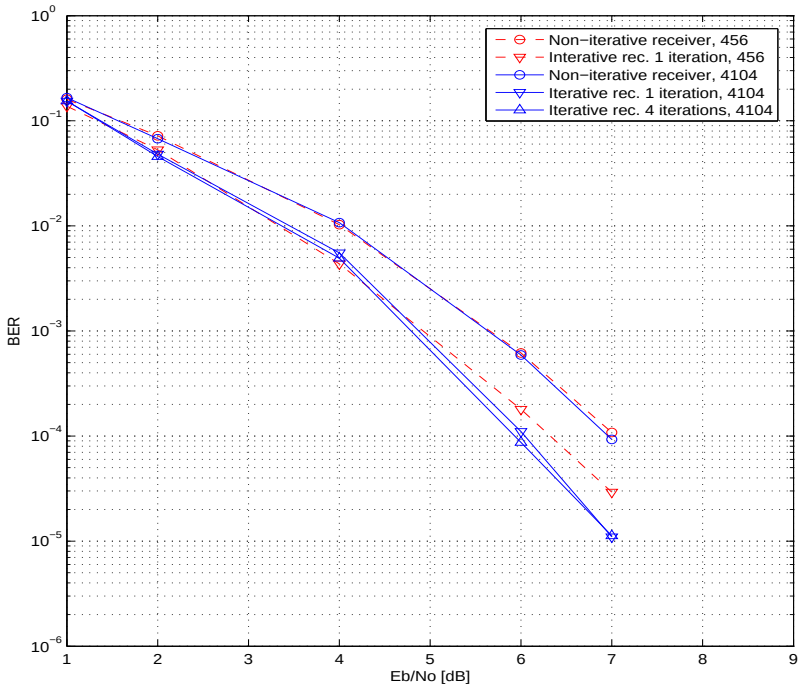


Figure 7.9: Performance of the iterative equalization and decoding with transmission through a 4-path quasi-static block fading channel with an interleaver size of 4104 bits instead of 456 bits.

The simulations has shown that E_b/N_0=5.6 dB for the 0'th iteration and 4.9 dB for the 1'st iteration at $BER = 10^{-3}$. This is the same result as for the interleaver size of 456 bits.
At $BER = 10^{-4}$, E_b/N_0=6.9 dB for the 0'th iteration and 6.0 dB for the 1'st iteration. For the small interleaver (456 bits) the results are E_b/N_0=6.9 dB for the 0'th iteration and 6.3 dB for the 1'st iteration.

The results in [Jordan et al., 1998], where they use a *Hilly Terrain* scenario, a quasi-static block fading channel, and a "pseudo random" interleaver of 4096 bits, shows that there is only little difference in the performance compared to the same setup but with an interleaver size of 456 bits. This result is similar to our result. They conclude that a larger interleaver can only be an advantage if the number of bursts also is increased due to fading in the channel.

7.4.3 Effect of Our Assumptions and Constraints

GMSK. In this project BPSK modulation is considered, while the GSM specifications considers GMSK modulation. GMSK itself can introduce ISI in the system.

Multiuser. Multiuser detection is not considered in this project. As time division is used in the GSM specifications, the effect of having several users has effect on the transmission rate, and it gives interference from other users at same frequency.

Hardware Signal Processing. The sampling process and other hardware signal processing processes are omitted in this project. A discrete time baseband representation of the transmitted signal is used. In physical systems, these hardware processes exist, thus adding noise to the system. Additive white Gaussian noise is incorporated in our system to represent thermal noise, but we should be aware that this is not a representation of all the noise from the hardware.

Channel Estimation. Instead of estimating the channel characteristics, perfect knowledge of the channel is assumed to be known in our project. This raises the performance of the receiver, as equalization always is based on estimated or known channel characteristics.

Time-Variant Channel Characteristics. The channel considered in this system is time-invariant for each transmitted burst. In real applications this is not the case. Equalization is not based on the optimal channel characteristics, thereby lowering the performance of the receiver.

Class 1 and 2 bits. According to GSM specifications, we should only encode part of the info word, but we encode the entire info word, and that must give better performance than that of a system true to the GSM specifications, as uncoded bits give worse performance than coded bits.

The "real" GSM Interleaver. The interleaver specified in the GSM specifications interleaves each code word together with the preceding and succeeding code words. This is not used in our project, but use of this interleaver would probably increase the performance, as the errors are distributed over several code words.

CHAPTER 8

Conclusion

Summary

The aim of this project is to improve a GSM receiver by using *iterative equalization and decoding*. Due to the time constraint we have restricted our system model to:

- A (23,33) convolutional channel encoder.
- An interleaver + burst assembler.
- A BPSK modulator (or mapper).
- A quasi-static tapped delay line channel model.
- An iterative equalizer and decoder.

The (23,33) convolutional encoder is the same as the one specified in the GSM standard, except that we do not distinguish $class$ 1, $class$ 2, and parity bits. We consider an incoming sequence, the info word, of 224 bits and add 4 tail bits. The encoded sequence, the code word, is of length 456 bits, as the GSM code word.

The considered interleaver is a modified version of the one specified in the GSM standard. Instead of interleaving one code word of 456 bits over 8 sequentially transmitted bursts in 16 half bursts of 57 bits, the modified interleaver interleaves one code word over only 4 sequentially transmitted bursts in 4 full bursts of 114 bits. By doing this, the interleaver blocks only contain code bits from one code word instead of two.

A tapped delay line model is used to model the time-variant multipath channel. We consider both fixed coefficients and Rayleigh distributed coefficients. The channel is quasi-static, meaning that the coefficients are time-invariant during one burst of 114 bits, and for each new burst the coefficients change.

A bank of matched filters are applied to the received signal. The output of the matched filters, which is four bursts (corresponding to one code word of 456 bits), is buffered. The equalizer, the deinterleaver, and the decoder is implemented in an iterative structure, where the equalizer and the decoder are exchanging *extrinsic information* with each other. Both the equalizer and the decoder are based on the Max-log-MAP Algorithm. Hard-decision is made on the output of the decoder after the last iteration.

Results

We will conclude on three performance results:

1. The considered model with Rayleigh fading channel. This will be the conclusion on our aim in this project.
2. The considered model with no-fading channel (fixed coefficients).
3. The considered model with Rayleigh fading channel, but where the interleaver size is increased from 456 bits to 4104 bits.

1. Simulations has shown that there is a performance gain from the 0'th iteration (which is the non-iterative receiver scheme) to the 1'st iteration from E_b/N_0 = 5.6 dB to 4.9 dB at $BER = 10^{-3}$. The following three iterations (2'nd, 3'rd, and 4'th iteration) do not give any appreciable performance gain. The results improve only very little compared to the 1'st iteration. At $BER = 10^{-2}$, E_b/N_0 is improved from 4.0 dB to 3.3 dB, and at $BER = 10^{-4}$, E_b/N_0 is improved from 7.0 dB to 6.3 dB. These results show that there is an improvement by using iterative equalization and decoding in the receiver in a GSM system, with the assumptions and constraints made in our system model.
It should be noted that it is only relevant to use the 0'th and the 1'st iteration. The rest are unnecessary, so to reduce complexity when considering implementation in an application, only the 0'th and the 1'st iteration should be considered.

2. An experiment, where the coefficients in the channel are fixed to $[\sqrt{0.45}, \sqrt{0.25}, \sqrt{0.15}, \sqrt{0.15}]$, has shown a performance gain for each new iteration from the 0'th iteration to the 4'th iteration. E_B/N_0 is 6.5 dB for the 0'th iteration and 4.5 dB for the 4'th iteration at $BER = 10^{-3}$. The same performance gain of approximately 2 dB is also achieved at $BER = 10^{-2}$ and $BER = 10^{-4}$.
This experiment is conducted to show, that the performance will increase for every new iteration from the 0'th iteration to the 4'th iteration, if we fix the coefficients so no fading will occur and none of the bursts are irrecoverable.

3. Another experiment, where the interleaver size is increased from 456 bits to 4104 bits has shown that E_b/N_0=5.6 dB for the 0'th iteration and 4.9 dB for the 1'st iteration at $BER = 10^{-3}$. This is the same result as for the interleaver size of 456 bits.
At $BER = 10^{-4}$, E_b/N_0=6.9 dB for the 0'th iteration and 6.0 dB for the 1'st iteration. For the small interleaver (456 bits) the results are E_b/N_0=6.9 dB for the 0'th iteration and 6.3 dB for the 1'st iteration.
From these results we can conclude, that there is a small improvement at high SNR, and there is no improvement at low SNR, when using an interleaver size of 4104 bits instead of 456 bits. We can also conclude, that in most cases it is not advisable to increase the interleaver size considering the rather modest performance gain archived compared to the increased time complexity.

Perspectives

If the *iterative equalization and decoding* scheme were to be implemented in a physical GSM system, some aspects were needed to be investigated:

- The "real" GSM interleaver and burst assembler should be implemented, and the performance should be tested.
- The GMSK modulation scheme should be implemented, and the performance should be tested.
- The code word structure, where only class 1 bits are encoded and class 2 bits are uncoded, should be implemented, and the performance should be tested.

Even though the above aspects will influence the performance of the iterative receiver, we would still expect an improvement in the performance, if the *iterative equalization and decoding* scheme were implemented in a GSM receiver.

Another interesting investigation would be to implement the *iterative equalization and decoding* scheme in an EDGE system, which is an upgrade of the GSM system. The difference between the GSM system and the EDGE system is the modulation scheme. EDGE uses 8-PSK instead of GMSK, and this would probably give a bigger gain in the performance due to the higher order modulation scheme.

APPENDIX A

Implementation

A block diagram of the implemented system is shown in figure A.1.

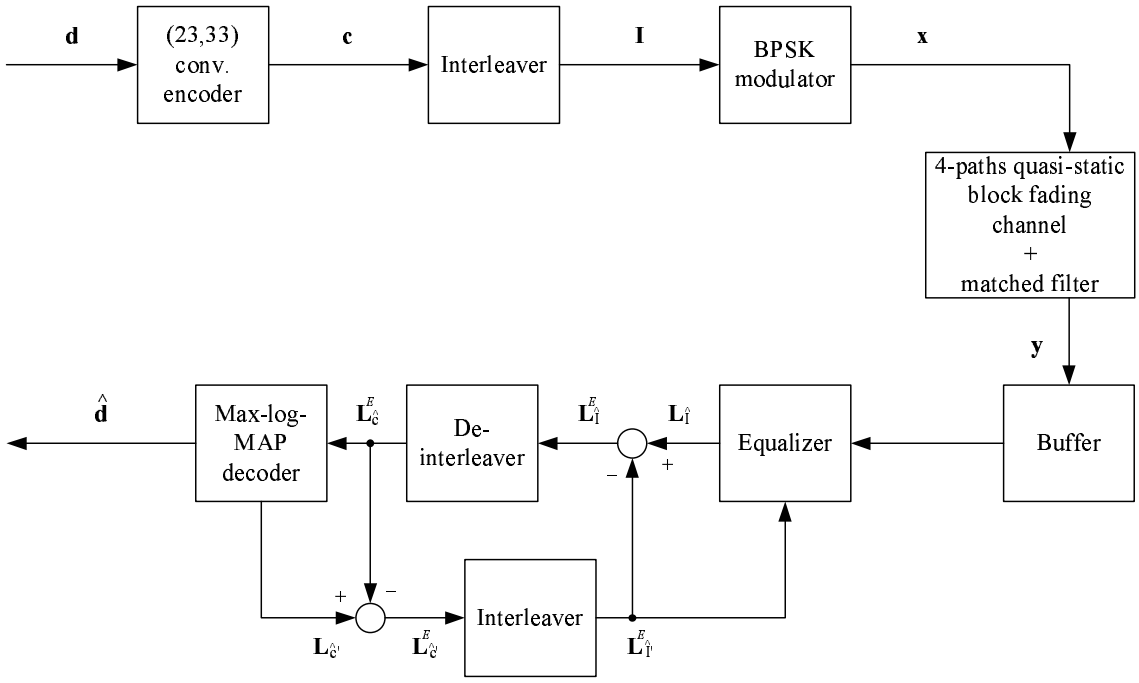


Figure A.1: Block diagram of the system model.

The flow of the system (corresponding to one realization) is the following:

1. Four different channel models, $\mathbf{a}^1$, $\mathbf{a}^2$, $\mathbf{a}^3$, and $\mathbf{a}^4$ (one for each burst), are generated with 4 Rayleigh distributed coefficients. Then AWGN is added and the matched filter is applied.
2. An info word, $\mathbf{d} \in \{0, 1\}$, containing 224 bits, is created randomly.
3. The info word is encoded, using the (23,33) convolutional encoder, into a code word, $\mathbf{c}$, of length 456.
4. The code word is interleaved and separated into 4 bursts, $\mathbf{I}^1$, $\mathbf{I}^2$, $\mathbf{I}^3$, and $\mathbf{I}^4$.

5. $\mathbf{I}^k$, where $k = \{1, 2, 3, 4\}$, is mapped onto a baseband representation of BPSK-symbols, $\mathbf{x}^k$.
6. $\mathbf{x}^k$ is, using channel model $\mathbf{a}^k$, distorted into the observed sequence $\mathbf{y}^k$.
7. $\mathbf{y}^k$ is equalized in the Max-log-MAP equalizer, resulting in a sequence of a priori information, $\mathbf{L}_{I^K}$.
8. $\mathbf{L}_{I^1}$, $\mathbf{L}_{I^2}$, $\mathbf{L}_{I^3}$, and $\mathbf{L}_{I^4}$ are deinterleaved and merged into the sequence $\mathbf{L}_c^P$ of length 456.
9. This sequence is decoded in the Max-log-MAP decoder, resulting in the sequence $\mathbf{L}_c$.
10. $\mathbf{L}_c^P$ is subtracted, giving the extrinsic information $\mathbf{L}_c^E$.
11. The interleaver interleaves and splits $\mathbf{L}_c^E$ into 4 bursts, $\mathbf{L}_{I^1}^P$, $\mathbf{L}_{I^2}^P$, $\mathbf{L}_{I^3}^P$, and $\mathbf{L}_{I^4}^P$, each of length 114.
12. $\mathbf{L}_{I^k}^P$, for $k = \{1, 2, 3, 4\}$, are, together with $\mathbf{y}^k$, used as input to the equalizer, resulting in new sequences of information, $\mathbf{L}_{I^k}$.
13. $\mathbf{L}_{I^k}^P$ is subtracted, giving the extrinsic information $\mathbf{L}_{I^k}^E$.
14. $\mathbf{L}_{I^1}^E$, $\mathbf{L}_{I^2}^E$, $\mathbf{L}_{I^3}^E$, and $\mathbf{L}_{I^4}^E$ are deinterleaved and merged into the sequence $\mathbf{L}_c^P$.
15. Steps 9-14 are repeated for each preceding iteration.
16. Finally hard decision is made based on the output of the decoder, resulting in an estimated info word, $\widehat{\mathbf{d}}$ of length 224.

Note that the buffer shown in the figure is not implemented as a separate function, but in the way that $\mathbf{y}^k$ for $k = \{1, 2, 3, 4\}$ is stored throughout all the iterations.

In the following the implementation of each block (except the buffer) in figure A.1 will be described.

A.1 (23,33) Convolutional Encoder

The purpose of the convolutional encoder is to encode an info word. The convolutional encoder is described in chapter 4

Input

- An info word, $\mathbf{d}$, of length m

Output

- A code word, $\mathbf{c}$, of length $2(m + 4)$

The encoder is initialized to zero-state, meaning $d_i = 0$ for $i \leq 0$.

Four zero-bits are added to the end of the info word to reset the encoder and terminate the trellis in the decoder.

$$\mathbf{d}^\bullet = \{d_1, d_2, ..., d_m, 0, 0, 0, 0\} \tag{A.1}$$

The encoder is, according the GSM standard, a rate $\frac{1}{2}$ convolutional encoder defined by the polynomials

$$g_1 = 1 + D^3 + D^4 \tag{A.2}$$
$$g_2 = 1 + D + D^3 + D^4 \tag{A.3}$$

It is implemented using the following equations

$$c_{2i-1} = d_i^\bullet \oplus d_{i-3}^\bullet \oplus d_{i-4}^\bullet \tag{A.4}$$
$$c_{2i} = d_i^\bullet \oplus d_{i-1}^\bullet \oplus d_{i-3}^\bullet \oplus d_{i-4}^\bullet \tag{A.5}$$

for $i = 1, 2, ..., m$.

A.2 Interleaver

The interleaver interleaves the code word and splits it up into 4 bursts.

Input

- A code word, **c**, of length 456.

Output

- Four bursts, $\mathbf{I}^1$, $\mathbf{I}^2$, $\mathbf{I}^3$, and $\mathbf{I}^4$, each of length 114.

The interleaver in GSM-systems is specified by

$$I_j^B = c_{m,k} \qquad \text{for } k = 1, 2, \ldots, 456 \tag{A.6}$$

where

$$m = 1, 2, \ldots \tag{A.7}$$
$$B = B_0 + 4m + ([k-1] \mod 8) \tag{A.8}$$
$$j = 2[(49[k-1]) \mod 57] + [([k-1] \mod 8) \text{ div } 4] \tag{A.9}$$

where B is the burst number, j is the bit number in burst B, m is the sequence number of the coded bits, and k is the bit number in the coded bits sequence m.

This GSM-specified interleaver interleaves each code word into 8 bursts, where the first 4 contain some of the preceding code word, and the last 4 also contain some of the succeeding code word. As this is unwanted due to the iterative receiver, some slight modifications have been made. Only one code word is considered, and it is being interleaved into only 4 bursts.

$$I_j^B = c_k \qquad \text{for } k = 1, 2, \ldots, 456 \tag{A.10}$$

where

$$B = ([k-1] \mod 4) + 1 \tag{A.11}$$
$$j = 2[(49[k-1]) \mod 57] + [([k-1] \mod 8) \text{ div } 4] \tag{A.12}$$

where $B \in \{1, 2, 3, 4\}$ is the burst number, $j \in \{1, 2, \ldots, 114\}$ is the bit number in burst B, and k is the bit number in the code word $\mathbf{c}$.

The outcome of interleaving the code word $\mathbf{c} = \{0, 1, 2, ..., 455\}$ is shown in table A.2 on the facing page.

A.3 BPSK Modulator

The purpose of the modulator is to map the code word into the corresponding baseband representation of BPSK. The function is described in chapter 3.

Input

- An interleaved code word, **I**, of length m

Output

- A baseband representation, $\mathbf{x}$, of length m

The modulator maps the received code bits to modulated symbols

$$x_i = \begin{cases} +1 & \text{for } I_i = 1 \\ -1 & \text{for } I_i = 0 \end{cases} \tag{A.13}$$

It is implemented using the following equation

$$x_i = (I_i - 0.5) \cdot 2 \qquad \text{for } i = 1, 2, ..., m \tag{A.14}$$

Burst j	1	2	3	4
0	0	57	114	171
1	228	285	342	399
2	64	121	178	235
3	292	349	406	7
4	128	185	242	299
5	356	413	14	71
6	196	249	306	363
7	420	21	78	135
8	256	313	370	427
9	28	85	142	199
10	320	377	434	35
11	92	149	206	263
12	384	441	42	99
13	156	213	270	327
14	448	49	106	163
15	220	277	334	391
16	56	113	170	227
17	284	341	398	455
18	120	177	234	291
19	348	405	6	63
20	184	241	298	355
21	412	13	70	127
22	248	305	362	419
23	20	77	134	191
24	312	369	426	27
25	84	141	198	255
26	376	433	34	91
27	148	205	262	319
28	440	41	98	155
29	212	269	326	383
30	48	105	162	219
31	276	333	390	447
32	112	169	226	283
33	340	397	454	55
34	176	233	290	347
35	404	5	62	119
36	240	297	354	411
37	12	69	126	183
38	304	361	418	19
39	76	133	190	247
40	368	425	26	83
41	140	197	254	311
42	432	33	90	147
43	204	261	318	375
44	40	97	154	211
45	268	325	382	439
46	104	161	218	275
47	332	389	446	47
48	168	225	282	339
49	396	453	54	111
50	232	289	346	403
51	4	61	118	175
52	296	353	410	11
53	68	125	182	239
54	360	417	18	75
55	132	189	246	303
56	424	25	85	139

Burst j	1	2	3	4
57	196	253	310	367
58	32	89	146	203
59	260	317	374	431
60	96	153	210	267
61	324	381	438	39
62	160	217	274	331
63	388	445	46	103
64	224	281	338	395
65	452	53	110	167
66	288	345	402	3
67	60	117	174	231
68	352	409	10	67
69	124	181	238	295
70	416	17	74	131
71	188	245	302	359
72	24	81	138	195
73	252	309	366	423
74	88	145	202	259
75	316	373	430	31
76	152	209	266	323
77	380	437	37	95
78	216	273	330	387
79	444	45	102	159
80	280	337	394	451
81	52	109	166	223
82	344	401	2	59
83	116	173	230	287
84	408	9	66	123
85	180	237	294	351
86	16	73	130	187
87	244	301	358	415
88	80	137	194	251
89	308	365	422	23
90	144	201	258	315
91	372	429	30	87
92	208	265	322	379
93	436	37	94	151
94	272	329	386	443
95	44	101	158	215
96	336	393	450	51
97	108	165	222	279
98	400	1	58	115
99	172	229	286	343
100	8	65	122	179
101	236	293	350	407
102	72	129	186	243
103	300	357	414	15
104	136	193	250	307
105	364	421	22	79
106	200	257	314	371
107	428	29	86	143
108	264	321	378	435
109	36	93	150	207
110	328	385	442	43
111	100	157	214	271
112	392	449	50	107
113	164	221	278	335

Figure A.2: The outcome of interleaving the code word $\mathbf{c} = \{0, 1, 2, ..., 455\}$.

A.4 4-Paths Quasi-static Block Fading Channel

The purpose of the channel is to distort the transmitted signal. The channel is described in chapter 2.

Input

- A baseband representation of BPSK-modulated symbols, $\mathbf{x}$, of length m.
- The channel coefficients, a_1, a_2, a_3 and a_4. Each a_l is generated as: $a_l = z_{l,1} + j \cdot z_{l,2}$, where $z_{l,1}$ and $z_{l,2}$ are realizations of a white Gaussian random process with zero-mean and variance 1.
- The signal-to-noise ratio, $\frac{E_b}{N_0}$.
- The code rate, R_c.

Output

- A baseband representation of distorted BPSK-modulated symbols, $\mathbf{y}$, of length m.

The channel is shown in figure A.3.

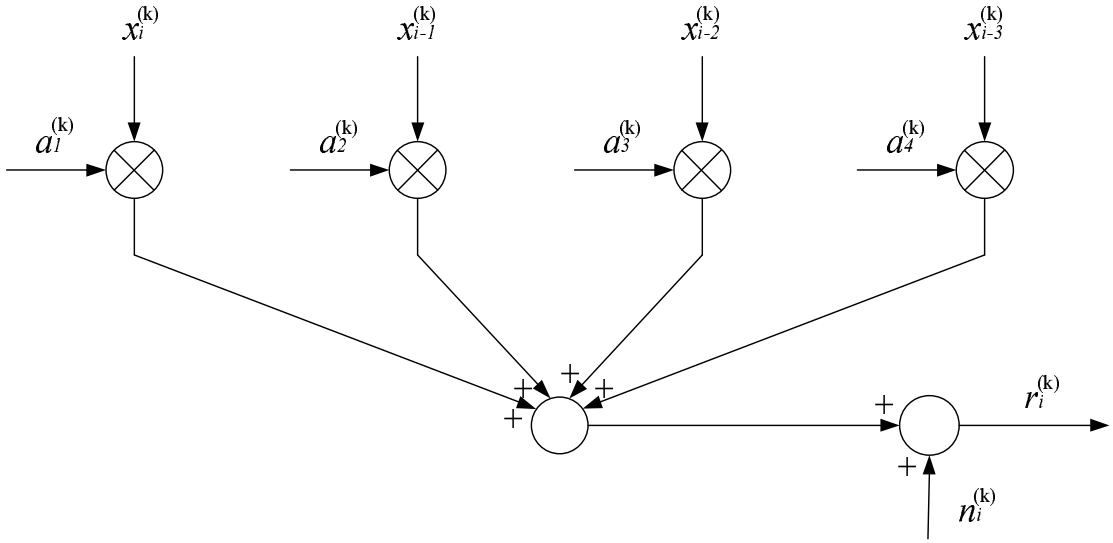


Figure A.3: Block diagram a 4-paths quasi-static block fading channel.

The channel and the corresponding matched filter is implemented as described in section 2.3.

$$\underline{y} \quad = \quad \underline{\underline{R}}\,\underline{x} + \underline{\underline{A}}^H\,\underline{n} \tag{A.15}$$

where $\underline{n}$ is a vector containing realizations of the white Gaussian noise variable N, $\underline{\underline{R}} = \underline{\underline{A}}^H \ \underline{\underline{A}}$ and

$$\underline{\underline{A}}^H = \begin{bmatrix} a_1^* & a_2^* & a_3^* & \cdots & a_L^* & 0 & 0 & \cdots \\ 0 & a_1^* & a_2^* & a_3^* & \cdots & a_L^* & 0 & \cdots \\ 0 & 0 & a_1^* & a_2^* & a_3^* & \cdots & a_L^* & \cdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix} \tag{A.16}$$

where a_1, a_2 etc. are channel taps. Note that the sequences $\mathbf{x}$ and $\mathbf{y}$ are represented as vectors.

A.4.1 Generating the Additive White Gaussian Noise

N can be generated as

$$N = \sigma_n \cdot (Z_{Re} + j \cdot Z_{Im}) \tag{A.17}$$

where σ_n is the standard deviation of the noise, and Z_{Re} and Z_{Im} are normal distributed random processes with variance 1 and zero mean.

The noise variance is dependent of the SNR, which can be expressed as

$$SNR = \frac{2E_s}{N_0} = 2R_c\frac{E_b}{N_0} \tag{A.18}$$

where E_s is the energy in each symbol after encoding, and R_c is the coding rate. The expression in equation A.18 is derived in subsection 2.3 on page 18.

SNR is also the ratio between the signal variance and the noise variance

$$SNR = \frac{\sigma_s^2}{\sigma_n^2} \tag{A.19}$$

and since $\sigma_s^2 = E[\mathbf{x}^2] = 1$, the noise variance can be derived from equation A.18 and A.19 as

$$SNR = 2R_c \cdot \frac{E_b}{N_0} = \frac{\sigma_s^2}{\sigma_n^2} \tag{A.20}$$

$$\Downarrow$$

$$\sigma_n^2 = \frac{1}{2R_c \cdot \frac{E_b}{N_0}} \tag{A.21}$$

The white Gaussian noise variable from equation A.17 is then described as

$$N = \sigma_n \cdot (Z_{Re} + j \cdot Z_{Im}) = \sqrt{\frac{1}{2R_c \cdot \frac{E_b}{N_0}}} \cdot (Z_{Re} + j \cdot Z_{Im}) \tag{A.22}$$

A.5 Equalizer

Input

- An observed sequence, $\mathbf{y}$, of length m
- An sequence of log-likelihood ratios, $\mathbf{L}_I^P$, from the decoder.

Output

- A sequence of log-likelihood ratios, $\mathbf{L}_I$, of length m

As the input, $\mathbf{L}_I^P$ is in the form of log-likelihood ratios, $\log \frac{P(c_i=1)}{P(c_i=0)}$, but log-probabilities are needed, the following equations are used

$$\log P(I_i = -1) = -\log\left(1 + e^{\log \frac{P(I_i=+1)}{P(I_i=-1)}}\right) \tag{A.23}$$

$$\log P(I_i = +1) = \log \frac{P(I_i = +1)}{P(I_i = -1)} + \log P(I_i = -1) \tag{A.24}$$

One stage of the trellis diagram used in the equalizer is shown in figure A.4.

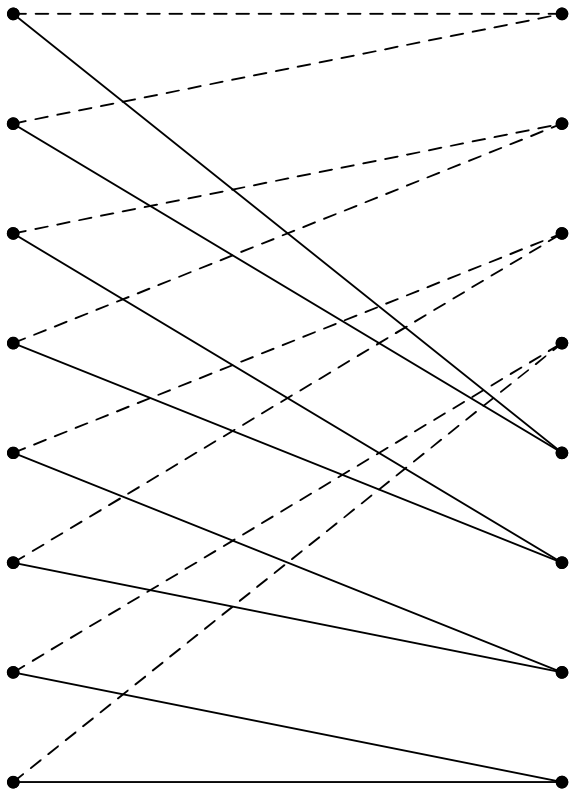


Figure A.4: One stage of the trellis used in the equalizer.

The following equations is derived in chapter 6 as the log-probability of a given branch in the trellis, depending on the observed sequence $\mathbf{y}$.

$$\gamma'_k(i) = -\sigma_n^{-1} \cdot Re(u_k \cdot (\Sigma_{j=1}^{i-1} \varrho_{i,j} \cdot u_j - y_i)) \tag{A.25}$$

This gives, using figure A.4, the following equations

$$\begin{aligned}
\gamma'_1(i) &= -\sigma_n^{-1} Re(+\varrho_{1,2} + \varrho_{1,3} + \varrho_{1,4} + y_i) + \log P(I_i = -1) && \text{(A.26)}\\
\gamma'_2(i) &= -\sigma_n^{-1} Re(-\varrho_{1,2} - \varrho_{1,3} - \varrho_{1,4} - y_i) + \log P(I_i = +1) && \text{(A.27)}\\
\gamma'_3(i) &= -\sigma_n^{-1} Re(+\varrho_{1,2} + \varrho_{1,3} - \varrho_{1,4} + y_i) + \log P(I_i = -1) && \text{(A.28)}\\
\gamma'_4(i) &= -\sigma_n^{-1} Re(-\varrho_{1,2} - \varrho_{1,3} + \varrho_{1,4} - y_i) + \log P(I_i = +1) && \text{(A.29)}\\
\gamma'_5(i) &= -\sigma_n^{-1} Re(+\varrho_{1,2} - \varrho_{1,3} + \varrho_{1,4} + y_i) + \log P(I_i = -1) && \text{(A.30)}\\
\gamma'_6(i) &= -\sigma_n^{-1} Re(-\varrho_{1,2} + \varrho_{1,3} - \varrho_{1,4} - y_i) + \log P(I_i = +1) && \text{(A.31)}\\
\gamma'_7(i) &= -\sigma_n^{-1} Re(+\varrho_{1,2} - \varrho_{1,3} - \varrho_{1,4} + y_i) + \log P(I_i = -1) && \text{(A.32)}\\
\gamma'_8(i) &= -\sigma_n^{-1} Re(-\varrho_{1,2} + \varrho_{1,3} + \varrho_{1,4} - y_i) + \log P(I_i = +1) && \text{(A.33)}\\
\gamma'_9(i) &= -\sigma_n^{-1} Re(-\varrho_{1,2} + \varrho_{1,3} + \varrho_{1,4} + y_i) + \log P(I_i = -1) && \text{(A.34)}\\
\gamma'_{10}(i) &= -\sigma_n^{-1} Re(+\varrho_{1,2} - \varrho_{1,3} - \varrho_{1,4} - y_i) + \log P(I_i = +1) && \text{(A.35)}\\
\gamma'_{11}(i) &= -\sigma_n^{-1} Re(-\varrho_{1,2} + \varrho_{1,3} - \varrho_{1,4} + y_i) + \log P(I_i = -1) && \text{(A.36)}\\
\gamma'_{12}(i) &= -\sigma_n^{-1} Re(+\varrho_{1,2} - \varrho_{1,3} + \varrho_{1,4} - y_i) + \log P(I_i = +1) && \text{(A.37)}\\
\gamma'_{13}(i) &= -\sigma_n^{-1} Re(-\varrho_{1,2} - \varrho_{1,3} + \varrho_{1,4} + y_i) + \log P(I_i = -1) && \text{(A.38)}\\
\gamma'_{14}(i) &= -\sigma_n^{-1} Re(+\varrho_{1,2} + \varrho_{1,3} - \varrho_{1,4} - y_i) + \log P(I_i = +1) && \text{(A.39)}\\
\gamma'_{15}(i) &= -\sigma_n^{-1} Re(-\varrho_{1,2} - \varrho_{1,3} - \varrho_{1,4} + y_i) + \log P(I_i = -1) && \text{(A.40)}\\
\gamma'_{16}(i) &= -\sigma_n^{-1} Re(+\varrho_{1,2} + \varrho_{1,3} + \varrho_{1,4} - y_i) + \log P(I_i = +1) && \text{(A.41)}
\end{aligned}$$

for $i = 1, 2, ..., m$

From section 6.4 we have, that

$$\alpha'_{u_k}(i) = \max_j \Big[\alpha'_{u_j}(i-1) + \gamma'_{u_j,u_k}(i-1)\Big] \tag{A.42}$$

$$\beta'_{u_k}(i) = \max_j \Big[\beta'_{u_j}(i+1) + \gamma'_{u_k,u_j}(i)\Big] \tag{A.43}$$

where u_k and u_j are states in the trellis.

This gives, when using figure A.4, the following equations

$$\begin{aligned}
\alpha'_1(i) &= \max[\alpha'_1(i-1) + \gamma'_1(i-1), \alpha'_2(i-1) + \gamma'_3(i-1)] && \text{(A.44)}\\
\alpha'_2(i) &= \max[\alpha'_3(i-1) + \gamma'_5(i-1), \alpha'_4(i-1) + \gamma'_7(i-1)] && \text{(A.45)}\\
\alpha'_3(i) &= \max[\alpha'_5(i-1) + \gamma'_9(i-1), \alpha'_6(i-1) + \gamma'_{11}(i-1)] && \text{(A.46)}\\
\alpha'_4(i) &= \max[\alpha'_7(i-1) + \gamma'_{13}(i-1), \alpha'_8(i-1) + \gamma'_{15}(i-1)] && \text{(A.47)}\\
\alpha'_5(i) &= \max[\alpha'_1(i-1) + \gamma'_2(i-1), \alpha'_2(i-1) + \gamma'_4(i-1)] && \text{(A.48)}\\
\alpha'_6(i) &= \max[\alpha'_3(i-1) + \gamma'_6(i-1), \alpha'_4(i-1) + \gamma'_8(i-1)] && \text{(A.49)}\\
\alpha'_7(i) &= \max[\alpha'_5(i-1) + \gamma'_{10}(i-1), \alpha'_6(i-1) + \gamma'_{12}(i-1)] && \text{(A.50)}\\
\alpha'_8(i) &= \max[\alpha'_7(i-1) + \gamma'_{14}(i-1), \alpha'_8(i-1) + \gamma'_{16}(i-1)] && \text{(A.51)}
\end{aligned}$$

for $i = 2, 3, ..., m+1$

$$\begin{aligned}
\beta'_1(i) &= \max[\beta'_1(i+1)+\gamma'_1(i), \beta'_5(i+1)+\gamma'_2(i)] && \text{(A.52)}\\
\beta'_2(i) &= \max[\beta'_1(i+1)+\gamma'_3(i), \beta'_5(i+1)+\gamma'_4(i)] && \text{(A.53)}\\
\beta'_3(i) &= \max[\beta'_2(i+1)+\gamma'_5(i), \beta'_6(i+1)+\gamma'_6(i)] && \text{(A.54)}\\
\beta'_4(i) &= \max[\beta'_2(i+1)+\gamma'_7(i), \beta'_6(i+1)+\gamma'_8(i)] && \text{(A.55)}\\
\beta'_5(i) &= \max[\beta'_3(i+1)+\gamma'_9(i), \beta'_7(i+1)+\gamma'_{10}(i)] && \text{(A.56)}\\
\beta'_6(i) &= \max[\beta'_3(i+1)+\gamma'_{11}(i), \beta'_7(i+1)+\gamma'_{12}(i)] && \text{(A.57)}\\
\beta'_7(i) &= \max[\beta'_4(i+1)+\gamma'_{13}(i), \beta'_8(i+1)+\gamma'_{14}(i)] && \text{(A.58)}\\
\beta'_8(i) &= \max[\beta'_4(i+1)+\gamma'_{15}(i), \beta'_8(i+1)+\gamma'_{16}(i)] && \text{(A.59)}
\end{aligned}$$

for $i = m, m-1, ..., 1$

Now that γ''s, α''s and β''s are known, we can find the log-probability, δ', of each branch, based on the entire sequence of code symbols.

$$\delta'_{u_k,u_j}(i) = \alpha'_{u_k}(i) + \gamma'_{u_k,u_j}(i) + \beta'_{u_j}(i+1) \quad \text{(A.60)}$$

This gives the following equations

$$\begin{aligned}
\delta_1(i) &= \max[\alpha'_1(i)+\gamma'_1(i)+\beta'_1(i+1), \alpha'_2(i)+\gamma'_3(i)+\beta'_1(i+1)] && \text{(A.61)}\\
\delta_2(i) &= \max[\alpha'_3(i)+\gamma'_5(i)+\beta'_2(i+1), \alpha'_4(i)+\gamma'_7(i)+\beta'_2(i+1)] && \text{(A.62)}\\
\delta_3(i) &= \max[\alpha'_5(i)+\gamma'_9(i)+\beta'_3(i+1), \alpha'_6(i)+\gamma'_{11}(i)+\beta'_3(i+1)] && \text{(A.63)}\\
\delta_4(i) &= \max[\alpha'_7(i)+\gamma'_{13}(i)+\beta'_4(i+1), \alpha'_8(i)+\gamma'_{15}(i)+\beta'_4(i+1)] && \text{(A.64)}\\
\delta_5(i) &= \max[\alpha'_1(i)+\gamma'_2(i)+\beta'_5(i+1), \alpha'_2(i)+\gamma'_4(i)+\beta'_5(i+1)] && \text{(A.65)}\\
\delta_6(i) &= \max[\alpha'_3(i)+\gamma'_6(i)+\beta'_6(i+1), \alpha'_4(i)+\gamma'_8(i)+\beta'_6(i+1)] && \text{(A.66)}\\
\delta_7(i) &= \max[\alpha'_5(i)+\gamma'_{10}(i)+\beta'_7(i+1), \alpha'_6(i)+\gamma'_{12}(i)+\beta'_7(i+1)] && \text{(A.67)}\\
\delta_8(i) &= \max[\alpha'_7(i)+\gamma'_{14}(i)+\beta'_8(i+1), \alpha'_8(i)+\gamma'_{16}(i)+\beta'_8(i+1)] && \text{(A.68)}
\end{aligned}$$

for $i = 1, 2, ..., m$

The output, $\mathbf{L}_I$, is calculated as

$$\mathbf{L}_I(i) = \max[\delta_5(i), \delta_6(i), \delta_7(i), \delta_8(i)] - \max[\delta_1(i), \delta_2(i), \delta_3(i), \delta_4(i)] \quad \text{(A.69)}$$

where $i = \{1, 2, ..., m\}$ and as the branches number $5, 6, 7,$ and 8 corresponds to $d_i = 1$, and branches number $1, 2, 3, 4$ corresponds to $d_i = 0$.

A.6 Deinterleaver

The purpose of the deinterleaver is to merge and deinterleave 4 burst into one code word. The function is the inverse of that described in section A.2.

Input

- 4 bursts, $\mathbf{I}_I^{E,1}$, $\mathbf{I}_I^{E,2}$, $\mathbf{I}_I^{E,3}$ and $\mathbf{I}_I^{E,4}$, each of length 114.

Output

- A code word, $\mathbf{L}_c^p$, of length 456.

The function is implemented using the following equation

$$c_k = I_j^B \qquad \text{for } k = 1, 2, \ldots, 456 \tag{A.70}$$

where

$$B = ([k-1] \mod 4) + 1 \tag{A.71}$$
$$j = 2[(49[k-1]) \mod 57] + [([k-1] \mod 8) \text{ div } 4] \tag{A.72}$$

where B is the burst number, j is the bit number in burst B, and k is the bit number in the code word.

A.7 Max-log-MAP Decoder

The purpose of the Max-log-MAP decoder is to decode the received code word. The function of a Viterbi decoder is described is chapter 5, and the Max-log-MAP algorithm is described is chapter 6.

Input

- Log-likelihood ratios $\mathbf{L}_c^P$ of a code word of length $2m + 8$

Output

- Log-likelihood ratios $\mathbf{L}_c$ of a code word of length $2m + 8$
- Log-likelihood ratios $\mathbf{L}^d$ of an info word of length m

The convolutional encoder consists of 4 memory elements. This leads to 16 states. The diagram for one stage in the Trellis diagram is shown in figure A.5.

As the input is in the form of log-likelihood ratios, $\log \frac{P(c_i=1)}{P(c_i=0)}$, but what is needed is log-probabilities, the following equations are used

$$\log P(c_i = 0) = -\log\left(1 + e^{\log \frac{P(c_i=1)}{P(c_i=0)}}\right) \tag{A.73}$$
$$\log P(c_i = 1) = \log \frac{P(c_i = 1)}{P(c_i = 0)} + \log P(c_i = 0) \tag{A.74}$$

In the Max-log-MAP algorithm, the branch metric, $\gamma_{k,j}(i)$, is the log-probability of the branch, Since we now have the probabilities of each of the symbols, corresponding to each branch, and we

consider these probabilities independent, the log-probability of a given branch is the sum of these log-probabilities.

$$\gamma'_1(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=0) \quad \text{(A.75)}$$
$$\gamma'_2(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=1) \quad \text{(A.76)}$$
$$\gamma'_3(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=1) \quad \text{(A.77)}$$
$$\gamma'_4(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=0) \quad \text{(A.78)}$$
$$\gamma'_5(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=1) \quad \text{(A.79)}$$
$$\gamma'_6(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=0) \quad \text{(A.80)}$$
$$\gamma'_7(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=0) \quad \text{(A.81)}$$
$$\gamma'_8(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=1) \quad \text{(A.82)}$$
$$\gamma'_9(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=0) \quad \text{(A.83)}$$
$$\gamma'_{10}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=1) \quad \text{(A.84)}$$
$$\gamma'_{11}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=1) \quad \text{(A.85)}$$
$$\gamma'_{12}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=0) \quad \text{(A.86)}$$
$$\gamma'_{13}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=1) \quad \text{(A.87)}$$
$$\gamma'_{14}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=0) \quad \text{(A.88)}$$
$$\gamma'_{15}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=0) \quad \text{(A.89)}$$
$$\gamma'_{16}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=1) \quad \text{(A.90)}$$
$$\gamma'_{17}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=1) \quad \text{(A.91)}$$
$$\gamma'_{18}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=0) \quad \text{(A.92)}$$
$$\gamma'_{19}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=0) \quad \text{(A.93)}$$
$$\gamma'_{20}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=1) \quad \text{(A.94)}$$
$$\gamma'_{21}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=0) \quad \text{(A.95)}$$
$$\gamma'_{22}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=1) \quad \text{(A.96)}$$
$$\gamma'_{23}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=1) \quad \text{(A.97)}$$
$$\gamma'_{24}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=0) \quad \text{(A.98)}$$
$$\gamma'_{25}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=1) \quad \text{(A.99)}$$
$$\gamma'_{26}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=0) \quad \text{(A.100)}$$
$$\gamma'_{27}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=0) \quad \text{(A.101)}$$
$$\gamma'_{28}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=1) \quad \text{(A.102)}$$
$$\gamma'_{29}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=0) \quad \text{(A.103)}$$
$$\gamma'_{30}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=1) \quad \text{(A.104)}$$
$$\gamma'_{31}(i) = \log P(c_{2i-1}=0) + \log P(c_{2i}=1) \quad \text{(A.105)}$$
$$\gamma'_{32}(i) = \log P(c_{2i-1}=1) + \log P(c_{2i}=0) \quad \text{(A.106)}$$

where $i = \{1, 2, ..., m+4\}$

From section 6.4 we have, that

$$\alpha'_{u_k}(i) = \max_j \Big[\alpha'_{u_j}(i-1) + \gamma'_{u_j,u_k}(i-1)\Big] \quad \text{(A.107)}$$

$$\beta'_{u_k}(i) = \max_j \Big[\beta'_{u_j}(i+1) + \gamma'_{u_k,u_j}(i)\Big] \quad \text{(A.108)}$$

where u_k and u_j are states in the trellis.

This gives, when using figure A.5, the following equations

$$\alpha'_1(i+1) = \max[\alpha'_1(i) + \gamma'_1(i), \alpha'_2(i) + \gamma'_3(i)] \quad \text{(A.109)}$$
$$\alpha'_2(i+1) = \max[\alpha'_3(i) + \gamma'_5(i), \alpha'_4(i) + \gamma'_7(i)] \quad \text{(A.110)}$$
$$\alpha'_3(i+1) = \max[\alpha'_5(i) + \gamma'_9(i), \alpha'_6(i) + \gamma'_{11}(i)] \quad \text{(A.111)}$$
$$\alpha'_4(i+1) = \max[\alpha'_7(i) + \gamma'_{13}(i), \alpha'_8(i) + \gamma'_{15}(i)] \quad \text{(A.112)}$$
$$\alpha'_5(i+1) = \max[\alpha'_9(i) + \gamma'_{17}(i), \alpha'_{10}(i) + \gamma'_{19}(i)] \quad \text{(A.113)}$$
$$\alpha'_6(i+1) = \max[\alpha'_{11}(i) + \gamma'_{21}(i), \alpha'_{12}(i) + \gamma'_{23}(i)] \quad \text{(A.114)}$$
$$\alpha'_7(i+1) = \max[\alpha'_{13}(i) + \gamma'_{25}(i), \alpha'_{14}(i) + \gamma'_{27}(i)] \quad \text{(A.115)}$$
$$\alpha'_8(i+1) = \max[\alpha'_{15}(i) + \gamma'_{29}(i), \alpha'_{16}(i) + \gamma'_{31}(i)] \quad \text{(A.116)}$$
$$\alpha'_9(i+1) = \max[\alpha'_1(i) + \gamma'_2(i), \alpha'_2(i) + \gamma'_4(i)] \quad \text{(A.117)}$$
$$\alpha'_{10}(i+1) = \max[\alpha'_3(i) + \gamma'_6(i), \alpha'_4(i) + \gamma'_8(i)] \quad \text{(A.118)}$$
$$\alpha'_{11}(i+1) = \max[\alpha'_5(i) + \gamma'_{10}(i), \alpha'_6(i) + \gamma'_{12}(i)] \quad \text{(A.119)}$$
$$\alpha'_{12}(i+1) = \max[\alpha'_7(i) + \gamma'_{14}(i), \alpha'_8(i) + \gamma'_{16}(i)] \quad \text{(A.120)}$$
$$\alpha'_{13}(i+1) = \max[\alpha'_9(i) + \gamma'_{18}(i), \alpha'_{10}(i) + \gamma'_{20}(i)] \quad \text{(A.121)}$$
$$\alpha'_{14}(i+1) = \max[\alpha'_{11}(i) + \gamma'_{22}(i), \alpha'_{12}(i) + \gamma'_{24}(i)] \quad \text{(A.122)}$$
$$\alpha'_{15}(i+1) = \max[\alpha'_{13}(i) + \gamma'_{26}(i), \alpha'_{14}(i) + \gamma'_{28}(i)] \quad \text{(A.123)}$$
$$\alpha'_{16}(i+1) = \max[\alpha'_{15}(i) + \gamma'_{30}(i), \alpha'_{16}(i) + \gamma'_{32}(i)] \quad \text{(A.124)}$$

where $i = \{1, 2, ..., m+4\}$

$$\beta'_1(i) = \max[\beta'_1(i+1)+\gamma'_1(i), \beta'_9(i+1)+\gamma'_2(i)] \quad \text{(A.125)}$$
$$\beta'_2(i) = \max[\beta'_1(i+1)+\gamma'_3(i), \beta'_9(i+1)+\gamma'_4(i)] \quad \text{(A.126)}$$
$$\beta'_3(i) = \max[\beta'_2(i+1)+\gamma'_5(i), \beta'_{10}(i+1)+\gamma'_6(i)] \quad \text{(A.127)}$$
$$\beta'_4(i) = \max[\beta'_2(i+1)+\gamma'_7(i), \beta'_{10}(i+1)+\gamma'_8(i)] \quad \text{(A.128)}$$
$$\beta'_5(i) = \max[\beta'_3(i+1)+\gamma'_9(i), \beta'_{11}(i+1)+\gamma'_{10}(i)] \quad \text{(A.129)}$$
$$\beta'_6(i) = \max[\beta'_3(i+1)+\gamma'_{11}(i), \beta'_{11}(i+1)+\gamma'_{12}(i)] \quad \text{(A.130)}$$
$$\beta'_7(i) = \max[\beta'_4(i+1)+\gamma'_{13}(i), \beta'_{12}(i+1)+\gamma'_{14}(i)] \quad \text{(A.131)}$$
$$\beta'_8(i) = \max[\beta'_4(i+1)+\gamma'_{15}(i), \beta'_{12}(i+1)+\gamma'_{16}(i)] \quad \text{(A.132)}$$
$$\beta'_9(i) = \max[\beta'_5(i+1)+\gamma'_{17}(i), \beta'_{13}(i+1)+\gamma'_{18}(i)] \quad \text{(A.133)}$$
$$\beta'_{10}(i) = \max[\beta'_5(i+1)+\gamma'_{19}(i), \beta'_{13}(i+1)+\gamma'_{20}(i)] \quad \text{(A.134)}$$
$$\beta'_{11}(i) = \max[\beta'_6(i+1)+\gamma'_{21}(i), \beta'_{14}(i+1)+\gamma'_{22}(i)] \quad \text{(A.135)}$$
$$\beta'_{12}(i) = \max[\beta'_6(i+1)+\gamma'_{23}(i), \beta'_{14}(i+1)+\gamma'_{24}(i)] \quad \text{(A.136)}$$
$$\beta'_{13}(i) = \max[\beta'_7(i+1)+\gamma'_{25}(i), \beta'_{15}(i+1)+\gamma'_{26}(i)] \quad \text{(A.137)}$$
$$\beta'_{14}(i) = \max[\beta'_7(i+1)+\gamma'_{27}(i), \beta'_{15}(i+1)+\gamma'_{28}(i)] \quad \text{(A.138)}$$
$$\beta'_{15}(i) = \max[\beta'_8(i+1)+\gamma'_{29}(i), \beta'_{16}(i+1)+\gamma'_{30}(i)] \quad \text{(A.139)}$$
$$\beta'_{16}(i) = \max[\beta'_8(i+1)+\gamma'_{31}(i), \beta'_{16}(i+1)+\gamma'_{32}(i)] \quad \text{(A.140)}$$

where $i = \{m+4, m+3, ..., 1\}$

Now that γ''s, α''s and β''s are known, we can find the log-probability, δ'_{u_k,u_j}, of each branch, based on the entire sequence of code symbols.

$$\delta'_{u_k,u_j}(i) = \alpha'_{u_k}(i) + \gamma'_{u_k,u_j}(i) + \beta'_{u_j}(i+1) \quad \text{(A.141)}$$

This gives the following equations

$$\delta'_1(i) = \alpha'_1(i) + \gamma'_1(i) + \beta'_1(i+1) \quad \text{(A.142)}$$
$$\delta'_2(i) = \alpha'_1(i) + \gamma'_2(i) + \beta'_9(i+1) \quad \text{(A.143)}$$
$$\delta'_3(i) = \alpha'_2(i) + \gamma'_3(i) + \beta'_1(i+1) \quad \text{(A.144)}$$
$$\delta'_4(i) = \alpha'_2(i) + \gamma'_4(i) + \beta'_9(i+1) \quad \text{(A.145)}$$
$$\delta'_5(i) = \alpha'_3(i) + \gamma'_5(i) + \beta'_2(i+1) \quad \text{(A.146)}$$
$$\delta'_6(i) = \alpha'_3(i) + \gamma'_6(i) + \beta'_{10}(i+1) \quad \text{(A.147)}$$
$$\delta'_7(i) = \alpha'_4(i) + \gamma'_7(i) + \beta'_2(i+1) \quad \text{(A.148)}$$
$$\delta'_8(i) = \alpha'_4(i) + \gamma'_8(i) + \beta'_{10}(i+1) \quad \text{(A.149)}$$
$$\delta'_9(i) = \alpha'_5(i) + \gamma'_9(i) + \beta'_3(i+1) \quad \text{(A.150)}$$
$$\delta'_{10}(i) = \alpha'_5(i) + \gamma'_{10}(i) + \beta'_{11}(i+1) \quad \text{(A.151)}$$
$$\delta'_{11}(i) = \alpha'_6(i) + \gamma'_{11}(i) + \beta'_3(i+1) \quad \text{(A.152)}$$
$$\delta'_{12}(i) = \alpha'_6(i) + \gamma'_{12}(i) + \beta'_{11}(i+1) \quad \text{(A.153)}$$
$$\delta'_{13}(i) = \alpha'_7(i) + \gamma'_{13}(i) + \beta'_4(i+1) \quad \text{(A.154)}$$
$$\delta'_{14}(i) = \alpha'_7(i) + \gamma'_{14}(i) + \beta'_{12}(i+1) \quad \text{(A.155)}$$
$$\delta'_{15}(i) = \alpha'_8(i) + \gamma'_{15}(i) + \beta'_4(i+1) \quad \text{(A.156)}$$
$$\delta'_{16}(i) = \alpha'_8(i) + \gamma'_{16}(i) + \beta'_{12}(i+1) \quad \text{(A.157)}$$
$$\delta'_{17}(i) = \alpha'_9(i) + \gamma'_{17}(i) + \beta'_5(i+1) \quad \text{(A.158)}$$
$$\delta'_{18}(i) = \alpha'_9(i) + \gamma'_{18}(i) + \beta'_{13}(i+1) \quad \text{(A.159)}$$
$$\delta'_{19}(i) = \alpha'_{10}(i) + \gamma'_{19}(i) + \beta'_5(i+1) \quad \text{(A.160)}$$
$$\delta'_{20}(i) = \alpha'_{10}(i) + \gamma'_{20}(i) + \beta'_{13}(i+1) \quad \text{(A.161)}$$
$$\delta'_{21}(i) = \alpha'_{11}(i) + \gamma'_{21}(i) + \beta'_6(i+1) \quad \text{(A.162)}$$
$$\delta'_{22}(i) = \alpha'_{11}(i) + \gamma'_{22}(i) + \beta'_{14}(i+1) \quad \text{(A.163)}$$
$$\delta'_{23}(i) = \alpha'_{12}(i) + \gamma'_{23}(i) + \beta'_6(i+1) \quad \text{(A.164)}$$
$$\delta'_{24}(i) = \alpha'_{12}(i) + \gamma'_{24}(i) + \beta'_{14}(i+1) \quad \text{(A.165)}$$
$$\delta'_{25}(i) = \alpha'_{13}(i) + \gamma'_{25}(i) + \beta'_7(i+1) \quad \text{(A.166)}$$
$$\delta'_{26}(i) = \alpha'_{13}(i) + \gamma'_{26}(i) + \beta'_{15}(i+1) \quad \text{(A.167)}$$
$$\delta'_{27}(i) = \alpha'_{14}(i) + \gamma'_{27}(i) + \beta'_7(i+1) \quad \text{(A.168)}$$
$$\delta'_{28}(i) = \alpha'_{14}(i) + \gamma'_{28}(i) + \beta'_{15}(i+1) \quad \text{(A.169)}$$
$$\delta'_{29}(i) = \alpha'_{15}(i) + \gamma'_{29}(i) + \beta'_8(i+1) \quad \text{(A.170)}$$
$$\delta'_{30}(i) = \alpha'_{15}(i) + \gamma'_{30}(i) + \beta'_{16}(i+1) \quad \text{(A.171)}$$
$$\delta'_{31}(i) = \alpha'_{16}(i) + \gamma'_{31}(i) + \beta'_8(i+1) \quad \text{(A.172)}$$
$$\delta'_{32}(i) = \alpha'_{16}(i) + \gamma'_{32}(i) + \beta'_{16}(i+1) \quad \text{(A.173)}$$

where $i = \{m+4, m+3, ..., 1\}$

The output, $\mathbf{L}^d$, is calculated as

$$L_i^d = \log P(d_i = 1) - \log P(d_i = 0) = \max_j[\delta'_j(i)] - \max_k[\delta'_k(i)] \quad \text{(A.174)}$$

where $i = \{1, 2, ..., m\}$ and

$$j = \{17, 18, ..., 32\} \quad \text{(A.175)}$$
$$k = \{1, 2, ..., 16\} \quad \text{(A.176)}$$

as the branches number j corresponds to $d_i = 1$, and branches number k corresponds to $d_i = 0$.

The output, $\mathbf{\Delta}$, is calculated as

$$\Delta_{2i-1} = \log P(c_{2i-1} = 1) - \log P(c_{2i-1} = 0) = \max_{j_1}[\delta'_{j_1}(i)] - \max_{k_1}[\delta'_{k_1}(i)] \quad \text{(A.177)}$$

$$\Delta_{2i} = \log P(c_{2i} = 1) - \log P(c_{2i} = 0) = \max_{j_2}[\delta'_{j_2}(i)] - \max_{k_2}[\delta'_{k_2}(i)] \quad \text{(A.178)}$$

where $i = \{1, 2, ..., m + 4\}$ and

$$j_1 = \{2, 3, 5, 8, 10, 11, 13, 16, 18, 19, 21, 24, 26, 27, 29, 32\} \quad \text{(A.179)}$$

$$k_1 = \{1, 4, 6, 7, 9, 12, 14, 15, 17, 20, 22, 23, 25, 28, 30, 31\} \quad \text{(A.180)}$$

$$j_2 = \{2, 3, 5, 8, 10, 11, 13, 16, 17, 20, 22, 23, 25, 28, 30, 31\} \quad \text{(A.181)}$$

$$k_2 = \{1, 4, 6, 7, 9, 12, 14, 15, 18, 19, 21, 24, 26, 27, 29, 32\} \quad \text{(A.182)}$$

as the branches number j_1 and k_1 corresponds to $c_{2i-1} = 1$ and $c_{2i-1} = 0$, respectively, and the branches number j_2 and k_2 corresponds to $c_{2i} = 1$ and $c_{2i} = 0$, respectively.

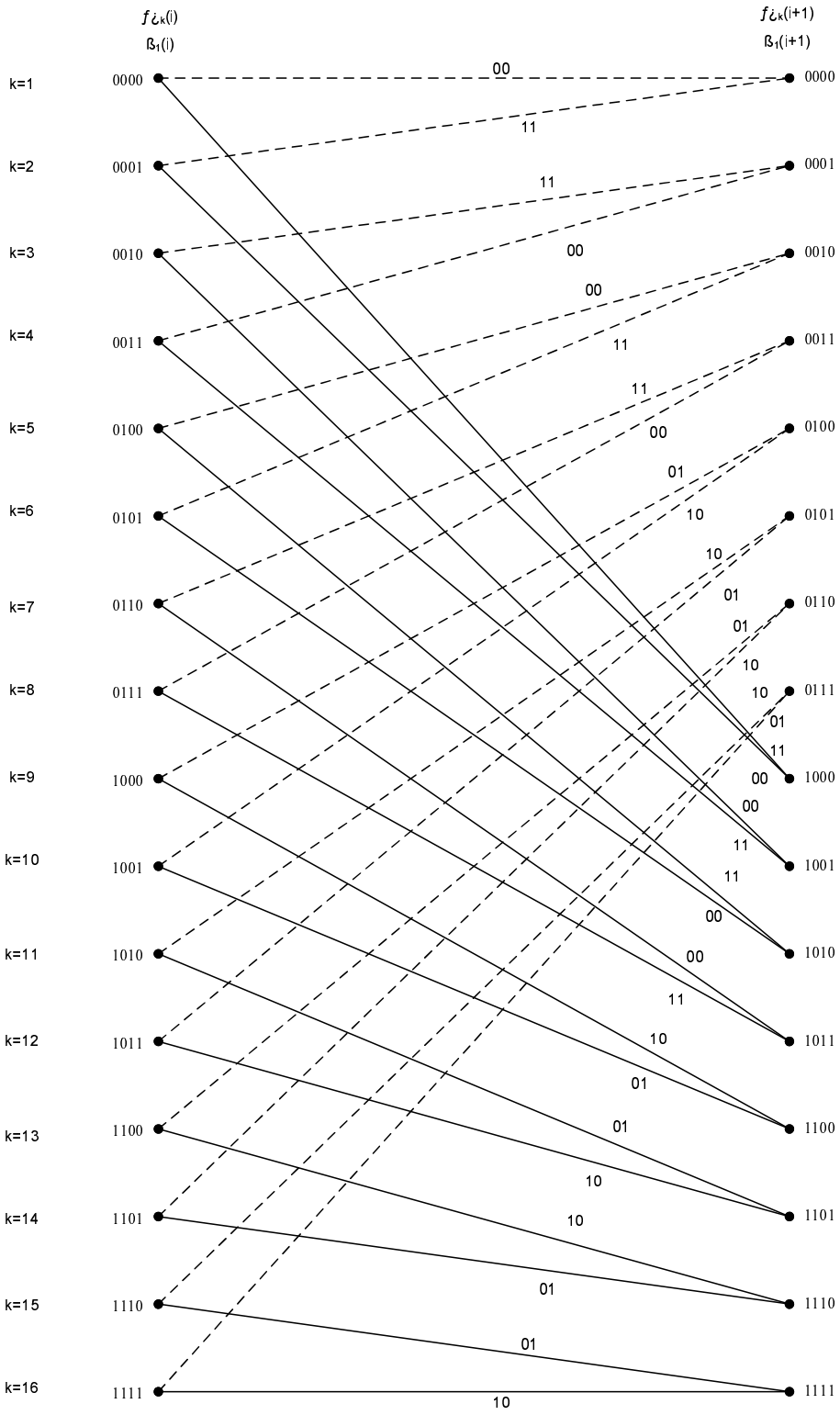


Figure A.5: One stage of the trellis.

Bibliography

[3GPP, 2004] 3GPP Organizational Partners (ARIB & ATIS & CCSA & ETSI & TTA & TTC). *3GPP TS 45.003 v4.3.0*. 3GGP, 2004.

[Bahl et al., 1974] L. R. Bahl & J. Cocke & F. Jelinek & and J. Raviv. *Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate*. IEEE Trans. Inform. Theory, vol. 20, pp. 284 287, 1974.

[Bauch et al., 1998] Gerhard Bauch & Volker Franz. *Iterative Equalization and Decoding for the GSM-System*. IEEE, 1998.

[Benedetto et al., 1998] Sergio Benedetto & Dariush Divsalar & Guido Montorsi & Fabrizio Pollara. *Serial Concatenation of Interleaved Codes: Performance Analysis, Design, and Iterative Decoding*. IEEE Transactions on Information Theory, 1998.

[Berrou et al., 1993] Claude Berrou & Alain Glavieux & Punya Thitimajshima. *Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-codes*. IEEE, 1993.

[ETSI, 2004] ETSI. *ETSI wins an Award for its Work on GSM Standard Development*, 2004-11. URL: `http://www.etsi.org/pressroom/previous/2000/gsmaward.htm`. Downloaded: 2005-02-24.

[Haykin, 2001] Simon Haykin. *Communication Systems*. John Wiley & Sons, 2001, 4th edition. ISBN: 0-471-17869-1.

[Hoeher, 1992] Peter Hoeher. *A Statistical Discrete-Time Model for the WSSUS Multipath Channel*. VTC, IEEE, 1992.

[Jordan et al., 1998] Frank Jordan & Kark-Dirk Kammeyer. *A Study on Iterative Decoding Techniques Applied to GSM Full-Rate Channels*. IEEE, 1998.

[Morelos-Zaragoza, 2002] Robert H. Morelos-Zaragoza. *The Art of Error Correcting Coding*. John Wiley & Sons, 2002.
ISBN: 0471495816.

[Proakis et al., 2002] John G. Proakis & Masoud Salehi. *Communication Systems Engeneering*. Prentice Hall, 2002, 2nd edition.
ISBN: 0-13-095007-6.