Signals & Systems

Spectral-Efficient Communication over Linear Gaussian Channels

an Information-Theoretic Approach

M.Sc. Thesis

Niek Johannes Bouman

University of Twente
Department of Electrical Engineering,
Mathematics & Computer Science (EEMCS)
Signals & Systems Group (SAS)
P.O. Box 217
7500 AE Enschede
The Netherlands

Report Number:	SAS23.07
Report Date:	October 18, 2007
Period of Work:	08/01/2007 – 30/10/2007
Thesis Committee:	Ir. H.S. (Harm) Cronie (Supervisor)
	Dr. ir. J.P.J. (Hans) Groenland
	Prof. dr. ir. C.H. (Kees) Slump
	Dr. ir. J.H. (Jos) Weber

Abstract

We present a capacity-approaching spectral-efficient block coded modulation scheme intended for communication over intersymbol interference (ISI) channels, based on multilevel coding and optimized irregular binary low-density parity-check (LDPC) component codes. Using superposition modulation, bits of multiple independent codewords are mapped to higher-order constellation symbols. This is done in a special way to achieve shaping gain. The outputs of multiple superposition modulators are interleaved to distribute the intersymbol interference over multiple independent codewords. We use multistage *a posteriori* probability (APP) detection and LDPC decoding, such that decoded bits are used as side information in the detection of subsequent levels.

We focus on the application of the scheme to the linear Gaussian channel, being a member of the class of ISI channels. We first discuss how to compute the capacity and a method for deriving a discrete-time channel model from a continuous-time channel transfer function. This discrete-time model is used in the APP detector as well as in a Monte Carlo algorithm to compute the achievable overall rate and the achievable rates per level. As an example, we design a communication system for the RC low-pass filter channel, for a scenario in which $E_s/N_0 = 7$ dB. Simulations show that the system, using optimized LDPC component codes with a blocklength of 10^5, operates reliably (a bit-error rate of around 10^{-5}) at 7.4 dB, at a rate equal to the water-pouring capacity at $E_s/N_0 = 6.0$ dB. Hence, for this particular scenario, the scheme operates reliably at only 1.4 dB away from capacity.

Preface

This report is the last piece of the big puzzle that represents my academic education at the University of Twente. Officially, this report concludes only the Master part of the education, but I regard it as the closure of the entire six-and-a-bit study years that I have spent in Enschede.

The Master programme consists of several courses, an internship and a thesis. One of the courses that I followed was entitled 'Information Theory', and was given by Fokke Hoeksema and Harm Cronie. This course remained one of my favourites until the end, and because of this course I developed great interest in information theory.

Via contacts of the Systems and Materials for Information Storage group, I got the opportunity to do an internship at the IBM Zurich Research Laboratory in Switzerland. There, I worked in the tape storage group, and learned about error-control coding for recording channels.

Once back in the Netherlands, I visited Harm to talk about the possibilities of a Master's project. The assignment description crystallized into a combination between Harm's current research and channels with memory, which I had encountered in IBM's tape group.

I have worked for nine months on this assignment with great pleasure and I am proud of the result. Particularly, I am enthusiastic about the fact that this work builds on two very recent publications (among them is one from my supervisor), which makes the work itself publishable, too.

Acknowledgements

I once read the following about writing acknowledgement sections in an article on report writing (Gerez, 1998): "Remember that it is the task of your supervisor to help you; you do not need to thank her/him for that." I will ignore this written piece of advice, and start by thanking Harm Cronie, my supervisor, for his pleasant style of supervision. Harm was always helpful and I have learned a lot from him.

I am grateful to Jos Weber (from Delft University of Technology) for receiving me in Delft and providing me with valuable feedback, and to Kees Slump (group head of the Signals and Systems group) and Hans Groenland (from the Systems and Materials for Information Storage group) for also constituting the graduation committee and reading and commenting on this 50-page thesis.

I wish to thank Henk and Mieke, my parents, for their unconditional support. And finally, I thank the many friends with whom I have had great times while having coffee or tea, beers, wines, dinners, barbecue-evenings, playing games like chess and poker, enjoying holidays, etc.

Table of Contents

Chapter 1

Introduction

We start this report by a layman's introduction into the project (intended for people without a background in mathematics or engineering). In the subsequent section, we explain what this thesis is actually about, and what problems we want to solve. Finally, we will discuss the structure of this thesis and the contents per chapter.

1.1 Layman's introduction

As the title indicates, this thesis deals with *communication*. But communication is not an unambiguous concept, and let us therefore start this thesis by giving our definition. We define communication as transportation of digital information. Digital information can be photos from your digital camera, a document, music, financial data, etcetera. Communication can take place between different locations in space, like mobile telephony. Another form of communication is storage of information (and retrieval of this information at a later point in time), for example using a hard disk. Both forms of communication are governed by the same information-theoretic principles. Additionally, these two types of communication bring out interesting relations between space and time. Transporting information from on location in space to another requires symbols to be transmitted sequentially in time. But to transport information through time, one needs physical space to store the information.

Although many parts of this monograph will be hard to follow for people without a background in electrical engineering or mathematics, the main concepts, however, should be easy to grasp for anyone. This comes from the fact that we, as human beings, are natural experts in communication. Often, people do not realize this, because all these communication principles are

hard-wired in our brains and seem so self-evident. Many parallels between human communication and digital communication can be found; we will show this by providing two examples.

1. Imagine that you are outside and having a conversation with someone. All of a sudden, the wind starts blowing. The person you are talking to asks you to speak up a bit, because he or she has problems understanding you. In technical terms, we would call the wind 'noise'. When the noise level increases, the so called 'signal-to-noise ratio' decreases. In order to maintain a constant signal-to-noise ratio, the signal power (in this case, the volume of your voice) should be increased.

2. A second, maybe more striking example is that of *language*. Humans communicate by making sentences consisting of words. And words consist of letters from an alphabet. Interestingly, we do not use all possible combinations of letters as words. This is not only because many of them would be unpronounceable. It is convenient to have some *distance* between words, to avoid misunderstandings.

   Nothing really new so far. But maybe you are surprised to hear that digital systems communicate nearly in the same way. They can only use a two-letter alphabet, namely zeros and ones. Like in our human language, not all possible combinations of the zeros and ones are used, but only a subset, called codewords. The idea is the same: the distance between the codewords allows for correct *decoding* of codewords that have been slightly altered during transport. One of the first commercial products in the Netherlands applying this so-called 'error-control coding' was Philips' Compact Disc.

In this thesis, a method is proposed to efficiently communicate at a rate that is as close as possible to the theoretical limit. To achieve this, state-of-the-art error-correcting codes (different from those applied in the Compact Disc system) are employed.

Research into more efficient communication schemes ultimately enables the development of better products, with respect to cost, (battery) power usage, speed, form factor, etcetera. Think of shorter upload times of photos from a digital camera to a computer (assuming that the data link between the camera and the PC forms the bottleneck) or the emergence of video telephony because of the vanishing costs of bandwidth.

1.2 Aim of the project

In many applications, reliable transport of digital information is required. With 'reliable', we mean that the received message will be identical to the transmitted message. In practice, communication channels are imperfect, in the sense that the received message is not always (or never) an exact

reproduction of the transmitted message. Another way of saying this is that the channel is *noisy*. Reliable communication over noisy channels can be achieved with *channel coding*.

In 1993, the field of channel coding made a big step forward by the invention of *Turbo codes*. These convolutional codes approached the ultimate Shannon capacity closer than ever before. Shortly after this invention, another breakthrough was made, namely the rediscovery of Gallager's low-density parity-check (LDPC) codes. These are large linear block codes, of which the parity-check matrix is sparse. Like Turbo codes, LDPC codes perform remarkably well, but they are easier to analyze. Around the year of 2000, methods have been found to optimize a binary LDPC code for a given channel. Although optimization schemes for non-binary LDPC codes exist as well, they yield fairly inaccurate results and are computationally very intensive. Moreover, decoding non-binary LDPC codes is more complex. A lot of today's research is focused on applying these and other state-of-the-art error-correcting codes to various channels.

In this project we will apply optimized binary LDPC codes to the linear Gaussian channel. The goal of the project is to achieve power- and bandwidth-efficient communication on this channel, with a scheme having acceptable computational complexity. We will assume a scenario in which the channel's characteristics are known at the transmitter.

The linear Gaussian channel model is quite general and has enough interesting properties to make this Master assignment challenging. The channel model possesses *memory*, meaning that the output of the channel depends not only on the current input, but also on past inputs. A consequent effect of channel memory is known as intersymbol interference. A typical example of a channel with memory is the binary-input magnetic recording channel, commonly known as the 'partial response channel'. We, however, do not restrict ourselves to binary-input channels. Hence, higher-order modulation can be used, which enables *spectral-efficient* communication at higher signal-to-noise ratios. We will apply *superposition modulation* with special constellation types that yield shaping gain, improving the power-efficiency of the transmitter. To be able to use binary codes on a non-binary channel, we will employ techniques called *multilevel coding* and *multistage decoding*. We will take an information-theoretic approach to the communication problem, alternative to conventional multicarrier transmission such as orthogonal frequency division multiplexing (OFDM).

1.3 Report organization

The structure of the report is as follows. Chapter 2 gives an introduction into low-density parity-check codes, and motivates the use of these codes in this project. Among other topics, the decoding algorithm is briefly discussed.

This chapter does not present new results.

Chapter 3 covers many aspects of the targeted linear Gaussian channel. The chapter is a collection of mainly existing results. It deals with the channel's information-theoretic properties, such as the capacity. A method is discussed to derive an equivalent discrete-time model from a description in the continuous-time domain. Moreover, an algorithm is given to compute a lower and upper bound on the achievable information rate.

In Chapter 4, a new communication scheme is proposed that is capable of approaching the theoretical capacity (derived in Chapter 3) in practice. Prior to presenting this scheme, we discuss two papers on which the scheme is primarily based, and we explain the important concept of multilevel coding in detail.

Most of the theory presented in this thesis is put into practice in Chapter 5. The proposed communication scheme is demonstrated on an RC low-pass network, which can be regarded as an instance of the linear Gaussian channel. Impatient readers that are hungry for results should jump to this chapter right away.

Chapter 6 concludes the report and gives some recommendations for further research.

Chapter 2

Low-Density Parity-Check Codes

This short chapter provides some basic information about low-density parity-check (LDPC) codes. After an introduction, we motivate the use of LDPC codes in this project. Subsequently, we discuss the 'sum-product' decoding algorithm. We explain the concept of coset randomization, which is, by the way, possible with every linear block code. We end by (only) touching the more advanced part of the theory on LDPC codes; code ensembles, concentration, asymptotic analysis (density evolution) and code design. The chapter does not contain new results.

2.1 Introduction

LDPC codes are linear block codes of which the parity-check matrix is sparse. They are decoded with a probabilistic decoding algorithm. LDPC codes were introduced by Gallager in his PhD thesis (1963), but somehow ignored and/or forgotten by the coding community, which was at that time more focused on algebraic codes, such as Reed-Solomon codes. In the nineties, shortly after the discovery of Turbo codes, LDPC codes were rediscovered by MacKay. Subsequently, the theory about encoding, analysis and design of LDPC codes has been expanded by contributions from, among others, Richardson, Urbanke, Shokrollahi and ten Brink. Since their rediscovery, LDPC codes belong to the state-of-the-art.

An LDPC code can be described by means of a Tanner graph. This is a bipartite graph, containing variable nodes, check nodes, and edges. An example of a bipartite graph is depicted in Figure 2.1. Decoding of LDPC codes works by running a message passing algorithm on the graph, we discuss

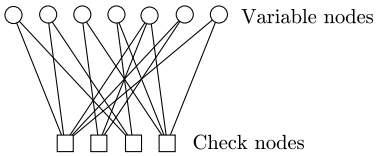


Figure 2.1: Example of a (very small) bipartite graph, in which the circles represent the variable nodes and the squares represent the check nodes (sometimes referred to as constraint nodes).

this algorithm in Section 2.3. The number of connections of a node is called the *degree* of that node. If the check node degrees and the variable node degrees are constant, the code is called *regular*. Otherwise, the code is called *irregular*, and we can speak about degree distributions. When we re-examine Figure 2.1, we can see that the depicted graph is irregular. Also, the LDPC codes that are employed in this project are irregular. It turns out that the degree distributions for the two node types determine many of the code's properties and decoding behavior.

2.2 Why (binary) LDPC codes are used in this project

In the communication scheme that we propose in Chapter 4, we use binary[1] LDPC codes, because of the following reasons.

1. The complexity of the decoder scales linearly in the blocklength. This is a very important property, since the longer the blocklength of an LDPC code, the better it performs (Gallager, 1968). In practice, the blocklength will be chosen depending on the maximum allowable delay.

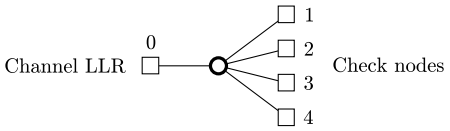
2. *Binary* LDPC codes can be very well optimized for the channel's statistics and can achieve capacity on the binary symmetric channel. Although optimization methods for non-binary LDPC codes (defined over a field of higher order) exist, they – by the time of this writing – do not work as well as the schemes for the binary case.

3. The decoder for non-binary LDPC codes is significantly more complex.

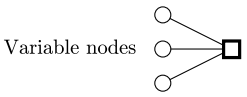
2.3 Decoding with the sum-product algorithm

Like other modern error correcting codes such as repeat-accumulate codes and fountain codes, LDPC codes can be efficiently decoded by 'message passing'. In message passing, each message contains an estimate of a transmitted bit. These messages are sent along the graph of the code. At each node, the messages are combined using a mathematical formula, known as the *update*

[1] defined over the Galois field $GF(2)$.



(a) At variable nodes, an outgoing message is computed as the sum of the other incoming messages. For example, $\ell_3 = \ell_0 + \ell_1 + \ell_2 + \ell_4$.



(b) At check nodes, the computation of an outgoing message involves hyperbolic tangent functions. The update formula is stated in Equation (2.2).

Figure 2.2: Update rules for variable nodes and check nodes.

rule. Messages are exchanged between nodes until enough information is at hand to decode the received data to a codeword.

The sum-product algorithm is a practical decoding algorithm for LDPC codes. It is not an optimal decoding method; it can only approximate the marginal posterior probabilities because of cycles that are present in the code graph. The optimal decoding algorithm would be maximum likelihood (ML) decoding, which is prohibitively complex. The sum-product algorithm makes extensive use of the distributive law (Aji and McEliece, 2000) to save computations.

In the implementation of the sum-product algorithm that we use , each message comprises a log-likelihood ratio (Richardson and Urbanke, 2007). The update rule for the variable nodes is very simple, an outgoing message is computed as the sum of all incoming messages, except the incoming message of the connection that is equal to the outgoing message,

$$\ell_j = \sum_{i \setminus j} \ell_i. \tag{2.1}$$

At check nodes, the update rule equals

$$\ell_j = 2 \tanh^{-1} \left(\prod_{i \setminus j} \tanh \left(\ell_i / 2 \right) \right). \tag{2.2}$$

Because of this latter expression, the form of the update rules presented here is sometimes called the 'hyperbolic tangent'-version of the update rules. See also Figure 2.2.

2.4 Simulating with the all-one codeword

Application of LDPC codes in practice requires not only a decoder but also an encoder. But for simulation purposes however, the encoder is usually left out of consideration since its behavior is straightforward and entirely

deterministic. The decoder behavior is the interesting part. In case of the additive white Gaussian noise (AWGN) channel, the common way to analyze the decoder is to assume transmission of the 'all-one' codeword. Note that the 'all-one codeword' (as also mentioned in title of this section) refers to the '+1'-element from the bipolar binary alphabet, $\{+1, -1\}$. If we would express the same codeword in terms of the Galois field $GF(2)$ elements, $\{0, 1\}$, we would call it the 'all-zero' codeword. This is because we use the following particular mapping:

$GF(2)$		Bipolar
0	$\leftrightarrow$	1
1	$\leftrightarrow$	-1

because of the elegant property that addition in $GF(2)$ corresponds to element-wise multiplication in the bipolar domain. The all-zero codeword (now speaking in terms of Galois field elements) will always exist, since LDPC codes are linear block codes.

2.4.1 Coset randomization

On the linear Gaussian channel, we cannot simply simulate with the all-one codeword; the effects of the channel memory would be inadequately simulated. To mimic realistic operation, the channel should be fed with fluctuating inputs to randomize the channel state. This could of course be achieved by just implementing the LDPC encoder. An easier way (that does not require an LDPC encoder) is to apply *coset randomization*. We start by explaining the notion of a *coset*.

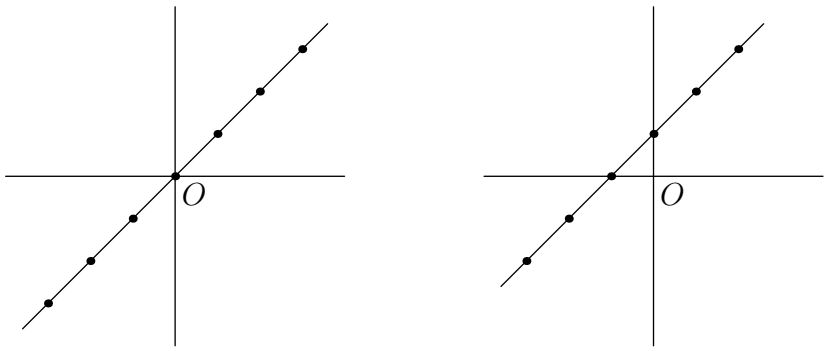
A linear code is usually a linear vector space V defined over a finite field, and fulfills the two following necessary conditions:

1. scalar multiples of codewords are codewords themselves;
2. adding two codewords yields another codeword.

Both conditions imply that the origin is contained (i.e. a linear code always contains the all-zero codeword). A coset is the *linear manifold* $V + u$; the space V that is possibly shifted away from the origin by the support vector u, as sketched in Figure 2.3. By adding a randomized u to the all-zero codeword at the transmitter side, the channel state is adequately randomized. At the receiver, we compensate for the added support vectors.

2.5 Aspects of code design

As mentioned in the introduction, the degree distributions for the variable nodes and check nodes play a key role in the performance and decodeability



(a) A linear code as a vector space over a finite field.

(b) A coset as a linear manifold.

Figure 2.3: Graphical interpretation of a linear code and one of its cosets.

of an LDPC code. There is a lot of theory behind this statement. This section will briefly mention some of this theory about analysis and design of LDPC codes. Although optimized LDPC codes are used in this project, the code optimization – which requires full understanding of the advanced theory about LDPC codes – was not done (and is not mastered) by the author of this thesis, but by the project's supervisor instead. An excellent reference for this theory is Richardson and Urbanke (2007).

2.5.1 Code ensembles and concentration

Degree distributions are probability distributions on the number of connections per node. From the variable node and check node degree distributions, a code can be generated (sampled) at random. For a given blocklength, the set of all possible code realizations generated from the same degree distributions pair is called an *ensemble*. It turns out, that every member of this ensemble has comparable behavior and performance. This fact, which can be proved mathematically, is termed *concentration*.

2.5.2 Density evolution

A way to gain insight into the asymptotic behavior of the decoding process of LDPC codes is to track the probability densities of log-likelihood messages (ℓ-densities) flowing through an infinite tree. This tree is an infinite-length cycle-free version of the bipartite code graph. This tracking process, termed *density evolution* by Richardson and Urbanke (2007), was already proposed by Gallager in his thesis. Density evolution can be implemented as an adapted version of the sum-product algorithm, in which the update rules operate on probability densities of log likelihood ratios, instead of on the log likelihood ratios themselves. With density evolution, the code's *thresh-*

old can be determined. The threshold is a parameter associated to a code with infinite blocklength that describes the maximum noise level at which decoding will still be successful.

2.5.3 Optimizing degree distributions

The way to optimize LDPC codes is to alter the degree distributions. The optimization criterion is the threshold: the higher the threshold, the better the code.[2] The threshold is computed with density evolution. Density evolution is computationally very complex and therefore time-consuming. To speed up the optimization process, the ℓ-densities are often approximated by Gaussian densities. This greatly reduces the computational complexity, since the node update rules only involve computations with means and variances. However, the Gaussian approximation is not very accurate. The state-of-the-art optimization schemes combine the unaccurate approximation and the costly but accurate density evolution method in a 'predictor-corrector' fashion, to achieve speed as well as accuracy.

[2] If the threshold is expressed as a signal-to-noise ratio, then it should be: "the lower the threshold, the better the code".

Chapter 3

The Linear Gaussian Channel

The Gaussian channel, named after its stochastic noise model, is a simple yet realistic channel model, adequate for many applications. The main goal of this chapter is to gain insight into the linear Gaussian channel, in which a linear filter is used to model the frequency-selectivity in the channel response. We investigate its information-theoretic properties and develop an equivalent discrete-time channel model. Finally, we study how the capacity and achievable information rates can be calculated from this discrete-time model. The chapter is a collection of mainly existing results.

3.1 Definition

The linear Gaussian channel is defined as

$$r(t) = g(t) * s(t) + n(t), \tag{3.1}$$

where the asterisk denotes the linear convolution operator, $g(t)$ is the impulse response of the channel, $s(t)$ is the continuous-time input signal and $n(t)$ is a zero-mean Gaussian noise process with power spectral density (PSD) $N(f)$.

3.2 Capacity of the linear Gaussian channel

The capacity, defined as the supremum of the mutual information, is the mutual information between input and output of the channel obtained by using the optimal input process. For the sake of simplicity, let us first consider the capacity of the ideal band-limited additive white Gaussian noise (AWGN) channel. The adjective 'ideal' implies that $g(t)$ has a flat frequency response and 'white' means that the PSD of $n(t)$ is flat.

3.2.1 Simplest case: the ideal AWGN channel

As we know from Shannon (1948), the capacity of the ideal AWGN channel is given by

$$C_{\text{ideal}} = W \log_2(1 + \text{SNR}) \text{ [bits/s]}, \tag{3.2}$$

where W is the one-sided bandwidth and SNR the signal-to-noise ratio. Equivalently, the capacity per unit bandwidth, the *spectral efficiency*, reads

$$\frac{C}{W}_{\text{ideal}} = \log_2(1 + \text{SNR}) \text{ [bits/s/Hz]}. \tag{3.3}$$

Sometimes it is more convenient to express the spectral efficiency in bits *per dimension* [bits/dim], especially in case of transmission of real-valued symbols. Like Proakis (2001), we therefore define the dimensionality parameter D as

$$D = \frac{d}{T} = 2W \text{ [dim/s]} \tag{3.4}$$

in which d is the number of dimensions and T is the symbol period. Using the dimensionality parameter, we can now write the spectral efficiency as

$$\frac{C}{D}_{\text{ideal}} = \frac{1}{2} \log_2(1 + \text{SNR}) \text{ [bits/dim]}. \tag{3.5}$$

Throughout the report, we will sometimes be a bit sloppy by using the word 'capacity' when we actually mean 'spectral efficiency'. Nevertheless, the right meaning can always be deducted from the specified unit of measure.

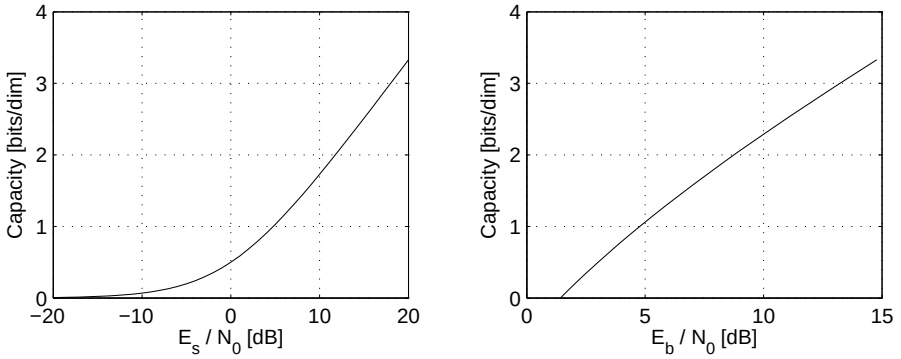
The SNR can be defined in different ways. A common form is E_s/N_0, in which E_s is the energy per symbol and N_0 the (one-sided) noise power spectral density. Another common form is the so-called *rate-compensated* signal-to-noise ratio E_b/N_0,

$$\frac{E_b}{N_0} = \frac{E_s}{RN_0}, \tag{3.6}$$

in which E_b denotes the energy per bit and R, the rate, has dimension bits per symbol. The capacity of the ideal Gaussian channel is plotted in Figure 3.1.

3.2.2 The optimal input process

To achieve capacity on additive Gaussian noise channels, the input probability density must be zero-mean Gaussian. This comes from the fact that, for a given variance, the Gaussian distribution maximizes the (differential) entropy. Apart from the probability density, the autocorrelation function of the input signal should be matched to the channel. In case of an ideal (flat frequency response) channel, the inputs should be uncorrelated. The exact relation between the filter response and the optimal input power spectral density will be treated more thoroughly later in this section.



(a) As a function of E_s/N_0.

(b) As a function of E_b/N_0. This plot was obtained by substituting $R = C/2W$ in Equation (3.6).

Figure 3.1: Different ways of plotting the capacity of the ideal (memoryless) Gaussian channel.

3.2.3 Capacity: the general case

Now that we have an understanding of the capacity concept and its dependence on bandwidth and SNR, we consider the general case, i.e. the capacity of the linear Gaussian channel. In this case, both the channel's magnitude response and the noise power spectral density may be frequency-dependent.

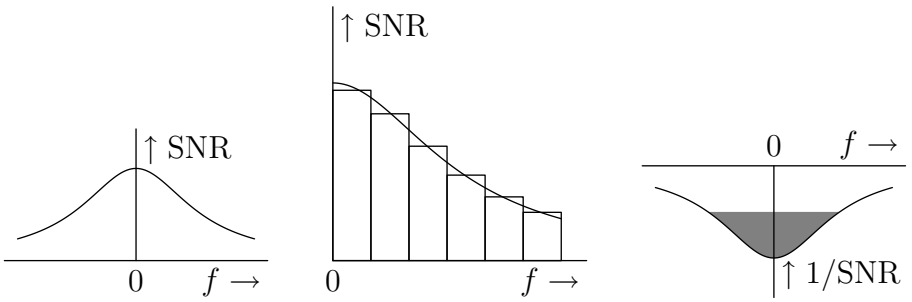
The capacity of the linear Gaussian channel can be found by dividing the continuous spectrum in infinitesimal intervals of bandwidth ϵ. In the limit for $\epsilon \to 0$, the noise PSD and channel response are flat in the band $[f_1, f_1 + \epsilon]$. In other words, in a frequency band of width ϵ, the channel is an ideal AWGN channel of which the capacity can be computed with Equation (3.2). This is illustrated in Figure 3.2–a,b.

Let us define a frequency-dependent channel SNR function as

$$\mathrm{SNR_c}(f) = \frac{|G(f)|^2}{N(f)}, \tag{3.7}$$

where $G(f)$ denotes the Fourier transform of $g(t)$, the channel impulse response. Under an input-power constraint, the optimal power allocation strategy is to concentrate the power in the regions where the channel SNR is high. This optimization is often termed *water-filling* or *water-pouring*, because it can be visualized as pouring water in the graph of the reciprocal of the channel SNR. This is shown in Figure 3.2–c. Note that water-pouring can only be applied in practice if the channel impulse response is known at the transmitter.

Forney and Ungerboeck (1998) present the following equations for the



(a) An arbitrary non-flat channel.

(b) Calculating the capacity of a non-flat channel by partitioning the channel into a set of narrow-band ideal channels.

(c) Finding the power spectral density of the optimal input process by water-pouring.

Figure 3.2: Capacity and input power allocation in case of a non-flat channel response.

capacity of the linear Gaussian channel, expressed in bits/s

$$C = \int_0^{f_{B,\text{edge}}} \log_2\left(1 + P_s^o(f)\text{SNR}_\text{c}(f)\right) \mathrm{d}f \tag{3.8}$$

$$= \int_0^{f_{B,\text{edge}}} \log_2\left[\frac{K|G(f)|^2}{N(f)}\right] \mathrm{d}f, \tag{3.9}$$

where $f_{B,\text{edge}}$ is the edge of the capacity-achieving band, $P_s^o(f)$ is the optimal input power spectral density and K is a constant that controls the power of the input signal. The capacity-achieving input power spectral density, which is dependent on this constant K, is given by

$$P_s^o(f) = \begin{cases} K - \dfrac{N(f)}{|G(f)|^2}, & |f| < f_{B,\text{edge}}, \\ \\ 0, & |f| \geq f_{B,\text{edge}}. \end{cases} \tag{3.10}$$

Note that from this relation, it follows that on the ideal AWGN channel (as mentioned before on page 12) the inputs should be uncorrelated to achieve capacity.

Let P be the available transmit power. $f_{B,\text{edge}}$ and K have to satisfy the power constraint

$$P = \int_{-f_{B,\text{edge}}}^{f_{B,\text{edge}}} K - \frac{N(f)}{|G(f)|^2} \mathrm{d}f. \tag{3.11}$$

Moreover, a continuity constraint is imposed on $P_s^o(f)$ at $f = f_{B,\text{edge}}$:

$$K - \frac{N(f_{B,\text{edge}})}{|G(f_{B,\text{edge}})|^2} = 0. \tag{3.12}$$

Equation (3.11) and (3.12) form a set of, in general, non-linear equations in which P, $N(f)$ and $|G(f)|^2$ are known, while K and $f_{B,\text{edge}}$ are the unknowns. In Chapter 5, we demonstrate for a particular example how to solve this set of equations.

3.3 Equivalent discrete-time channel model

To be able to use digital signal processing, a discrete-time version of the channel model is required. Forney and Ungerboeck (1998) give an in-depth treatment of computing an equivalent digital *finite* impulse response (FIR) filter from a continuous-time channel. Paradoxically, the conversion is optimal for an *infinite* number of filter coefficients. In this section we mention the important steps of the conversion method. We closely follow the notation of Forney and Ungerboeck (1998), which should ease the lookup of details in the reference. Instead of focusing on complex passband transmission, we assume baseband transmission of real-valued symbols. This is because in Chapter 5, we will demonstrate the theory by means of a practical channel example (an *RC* network) that is typically a baseband channel. However, it does not mean that the principles presented in this thesis cannot be used with complex passband transmission. Furthermore, we use two-sided frequency spectra and power spectral densities, whereas Forney and Ungerboeck decided to work with one-sided spectra. This is just a matter of choice.

We start by considering the desired capacity-achieving continuous-time input signal $s(t)$

$$s(t) = \sum_i a_i g_T(t - iT). \tag{3.13}$$

The information sequence $\{a_i\}$ should consist of i.i.d. real-valued Gaussian random variables. Digital information sources typically emit uniformly distributed bits, since the binary entropy function is maximized for $p(0) = p(1) = 1/2$. An elegant method to generate the sequence $\{a_i\}$ from Bernoulli-1/2 distributed bits is presented in Cronie (2007) and is discussed in Chapter 4.

In Equation (3.13), $g_T(t)$ is the symbol response of the transmit filter, that should shape the flat input spectrum to the optimal water-pouring spectrum. To be precise, the desired magnitude response of the transmit filter is a scaled version of $P_s^o(f)$

$$|G_T(f)|^2 = \frac{T}{\sigma_a^2} P_s^o(f), \tag{3.14}$$

where σ_a^2 is the variance per symbol of the sequence $\{a_i\}$.

At the receiver, the received signal is first passed through a noise-whitening filter. The Fourier transform of this filter is denoted by $G_w(f)$. The transfer

function from the transmitter to the receiver is

$$V(f) = G_T(f)G(f)G_w(f). \tag{3.15}$$

After the noise-whitening filter, a *matched filter* is employed, having a response $V^*(f)$ (the superscripted asterisk denotes the complex conjugate). The Fourier transform of the end-to-end symbol response $q(t)$ is

$$Q(f) = V(f)V^*(f) = \frac{|G_T(f)G(f)|^2}{N(f)} = \frac{T}{\sigma_a^2} P_s^o(f)\mathrm{SNR_c}(f). \tag{3.16}$$

In the receiver, $q(t)$ is sampled every T seconds, yielding a discrete sequence $\{q(\ell T)\}$ with discrete-time Fourier transform

$$\tilde{Q}(f) = \frac{1}{T} \sum_{m=-\infty}^{+\infty} Q(f + m/T). \tag{3.17}$$

The sequence $\{q(\ell T)\}$ can be used to form a discrete model in D-transform notation (where $D = z^{-1}$, we do not mean the dimensionality parameter D here)

$$\bar{y}(D) = a(D)q(D) + n(D), \tag{3.18}$$

but the disadvantage of this model is that the noise $n(D)$ is non-white, i.e. its autocorrelation sequence is equal to $\{q(\ell T)\}$.

3.3.1 An alternative 'canonical' discrete-time model

Using discrete-time spectral factorization (Papoulis, 1977), an alternative model is developed that not only has white Gaussian noise, but also a causal, monic (the zero-th order coefficient equals one) and minimum-phase response. Because of the latter three properties, the response is called 'canonical'. For the derivation of this factorization, refer to Forney and Ungerboeck (1998). The result is a recursive expression for the coefficients h_k of the discrete filter:

$$\begin{aligned} h_0 &= 1 \\ h_k &= \sum_{i=0}^{k-1} \frac{k-i}{k} h_i \alpha_{k-i} \qquad k \geq 1, \end{aligned} \tag{3.19}$$

where α are the coefficients of a Fourier series of $\log \tilde{Q}(f)$:

$$\alpha_\ell = T \int_{\langle 1/T \rangle} \log\left[\tilde{Q}(f)\right] \exp(\mathrm{j}2\pi f \ell T)\mathrm{d}f. \tag{3.20}$$

The *equivalent canonical discrete-time Gaussian channel* is

$$y(D) = a(D)h(D) + w(D), \tag{3.21}$$

where $w(D)$ is zero-mean Gaussian and has variance $1/A^2$, where A^2 follows from

$$\log A^2 = \alpha_0. \tag{3.22}$$

In order to limit the computational complexity of certain digital signal processing operations, the response h_k has to be truncated to a moderate length. If, after truncation, $\{h_k\}$ has length $M+1$, we call M the *memory length* of the equivalent discrete-time channel model.

In the time domain, we have arrived at the following model

$$y_n = \sum_{k=0}^{M} h_k a_{n-k} + w_n. \tag{3.23}$$

To get more insight into expressions such as Equation (3.19) and Equation (3.20) and the general process of deriving an equivalent discrete-time channel model, it is instructive to consider an example. We therefore refer the reader to Chapter 5, where these expressions are applied on a continuous-time transfer function of an electrical RC low-pass filter network.

3.3.2 The channel's conditional probability density function

In Equation (3.23), w_n is a realization of the white Gaussian noise process $\mathsf{W} \sim \mathcal{N}\left(0, \sigma^2\right)$, where $\sigma^2 = A^{-2}$. The probability density function of W is equal to

$$p_{\mathsf{W}}(w_n) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{{w_n}^2}{2\sigma^2}\right). \tag{3.24}$$

In order to find the conditional probability density function of Y, we examine Equation (3.23). The output y_n is a sum between the noise and the linear filter output. Let us denote the (noise-free) linear filter output as v_n. The mapping from the a_i to v_n is deterministic, hence the probability density of V conditional on A is a Dirac delta function:

$$p_{\mathsf{V}|\mathsf{A}}(v_n|a_{n-M}, \cdots, a_n) = \delta\left(v_n - \sum_{k=0}^{M} h_k a_{n-k}\right). \tag{3.25}$$

We find the conditional probability density of Y by convolving the noise density with the conditional density of the filter output:

$$\begin{aligned} p_{\mathsf{Y}|\mathsf{A}}(y_n|a_{n-M}, \cdots, a_n) &= \int_{-\infty}^{\infty} \mathrm{d}\zeta \; p_{\mathsf{W}}(\zeta) \; p_{\mathsf{V}|\mathsf{A}}(y_n - \zeta|a_{n-M}, \cdots, a_n), \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\left(y_n - \sum_{k=0}^{M} h_k a_{n-k}\right)^2}{2\sigma^2}\right), \end{aligned} \tag{3.26}$$

where the last step follows from the sifting property of the delta function (Soliman and Srinath, 1998).

We can express Equation (3.26) in a more compact form by switching to vector notation:

$$\begin{aligned} \mathbf{h} &= [h_0, h_1, \cdots, h_M]^{\mathrm{T}}, && (3.27)\\ \mathbf{a}_n &= [a_n, a_{n-1}, \cdots, a_{n-M}]^{\mathrm{T}}, && (3.28)\\ p_{\mathsf{Y}|\mathsf{A}}(y_n|\mathbf{a}_n) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\left(y_n - \mathbf{h}^{\mathrm{T}}\mathbf{a}_n\right)^2}{2\sigma^2}\right). && (3.29) \end{aligned}$$

3.4 Capacity of the equivalent discrete-time channel

The capacity of the equivalent discrete-time channel model should of course be the same as the capacity of the continuous-time model. But, when a truncated version of the impulse response is used, the capacity computed from the discrete model is expected to deviate from the continuous-time 'true' capacity.

The capacity of the discrete-time channel is derived in the paper of Hirt and Massey (1988). However, we have to be careful with their results, because Hirt and Massey (1988) assume a discrete-time channel model which does not incorporate a transmit filter. Hence, they employ water-pouring in the discrete-time Fourier domain. Applying their capacity formulas would be in essence applying water-pouring twice, which will yield incorrect results.

Instead, we will derive the capacity based on the discrete-time model from Equation (3.8). As in Hirt and Massey (1988), we define the discrete spectrum $H(\lambda)$ as the discrete-time Fourier transform (DTFT) of the channel impulse response

$$H(\lambda) = \sum_{k=0}^{M} h_k \exp(-\mathrm{j}k\lambda). \tag{3.30}$$

Let us consider the expression for the capacity based on the continuous-time model

$$\begin{aligned} C &= \int_0^{f_{B,\mathrm{edge}}} \log_2\left(1 + P_s^o(f)\mathrm{SNR}_{\mathrm{c}}(f)\right) \mathrm{d}f \\ &= \int_0^{f_{B,\mathrm{edge}}} \log_2\left(1 + \tilde{Q}(f)\right) \mathrm{d}f \ \text{[bits/s]}. && (3.31) \end{aligned}$$

The continuous-time frequency f and the discrete frequency λ are related as follows:

$$f = \frac{\lambda W}{\pi} = \frac{\lambda f_{B,\mathrm{edge}}}{\pi}. \tag{3.32}$$

We replace $\tilde{Q}(f)$ by its discrete-time equivalent $A^2|H(\lambda)|^2$, in which A^2 is the DC gain factor from the spectral factorization, defined in Equation (3.22).

Simultaneously, a change of variables is made by substituting the right hand side of Equation (3.32) into the infinitesimal $\mathrm{d}f$, and rewriting the bounds of integration using

$$\lambda = \frac{f\pi}{f_{B,\mathrm{edge}}}. \tag{3.33}$$

We obtain

$$C = \int_0^{\pi} \log_2 \left[1 + A^2|H(\lambda)|^2\right] \mathrm{d}\frac{\lambda f_{B,\mathrm{edge}}}{\pi}. \tag{3.34}$$

Finally, we bring the constants in the infinitesimal to the left and we divide by $2f_{B,\mathrm{edge}}$ (remember the dimensionality parameter $D = 2W$ introduced on page 12), to obtain a bandwidth-efficiency specified in bits per dimension:

$$\frac{C}{D}_{\mathrm{discrete-time}} = \frac{1}{2\pi} \int_0^{\pi} \log_2 \left[1 + A^2|H(\lambda)|^2\right] \mathrm{d}\lambda \text{ [bits/dim]}. \tag{3.35}$$

To our knowledge, this is a new (but not a spectacular) result.

3.5 A Monte Carlo method for estimating the information rate

The results regarding the capacity of the linear Gaussian channel that we obtained so far, hold under the assumption of a Gaussian input density. In practice, usually finite signal constellations are used. Even using superposition modulation, we can only approximate a Gaussian density. Therefore, we will now discuss a Monte Carlo method to estimate the information rate, given an arbitrary input.

The method, that was developed independently by Pfister et al. (2001), Arnold and Loeliger (2001) and Sharma and Singh (2001), relies on the asymptotic equipartition property (AEP) for continuous random variables (Cover and Thomas, 2006). Applied to our channel, the AEP states that the probability of a length-n sequence $\mathbf{y}^n$ drawn from a stationary ergodic finite-state Markov process Y is close to $2^{-n\mathrm{H}(\mathsf{Y})}$, i.e.:

$$\lim_{n\to\infty} \frac{1}{n} \log \frac{1}{p(\mathbf{y}^n)} \overset{\text{w.p. } 1}{=} \mathrm{H}(\mathsf{Y}). \tag{3.36}$$

The probability of such a sequence can be efficiently computed on the trellis using the forward recursion of the BCJR algorithm (Bahl et al., 1974). Details about the Monte Carlo algorithm can be be found in one of the three papers mentioned above, or in a nice overview of Yang and Kavčić (2005). Together with the entropy of Y given the input A, $\mathrm{H}(\mathsf{Y}|\mathsf{A})$, the mutual information is computed as

$$\mathrm{I}(\mathsf{A};\mathsf{Y}) = \mathrm{H}(\mathsf{Y}) - \mathrm{H}(\mathsf{Y}|\mathsf{A}). \tag{3.37}$$

3.5.1 Bounds on the information rate for the truncated channel model

The complexity of the BCJR algorithm is exponentially related to the length of the discrete-time channel impulse response. Therefore, we are forced to use a truncated version (in practice: not more than around four coefficients) of the discrete-time canonical model. The minimum-phase property of the canonical model makes that the energy is concentrated in the leading filter coefficients. Hence, truncating the low energy tail will probably have minor consequences. Nevertheless, we want quantify the relation between the truncation length and the accuracy of the information rate estimate. This can be done by making use of the lower and upper bound on the information rate discussed in Arnold et al. (2002). The idea is to sample a very long sequence of the output of a very accurate version of the canonical model (many coefficients), $\mathbf{y}^n$, given an arbitrary input vector $\mathbf{a}^n$. Subsequently, the BCJR algorithm – based on the truncated channel model that has only a few coefficients – is run on the output sequence, estimating $\mathrm{H}(\mathsf{Y})$.

The upper bound is given by

$$\hat{\mathrm{I}}_{\mathrm{upper}}(\mathsf{A};\mathsf{Y}) = \hat{\mathrm{H}}(\mathsf{Y}) - \mathrm{H}(\mathsf{W}). \tag{3.38}$$

in which $\mathrm{H}(\mathsf{W})$ is the differential entropy of the Gaussian noise. It can be be found analytically:

$$\begin{aligned} \mathrm{H}(\mathsf{W}) &= -\int_{S_{\mathsf{W}}} p_{\mathsf{W}}(w) \log p_{\mathsf{W}}(w) \mathrm{d}w & (3.39)\\ &= \frac{1}{2} \log_2 2\pi e\sigma^2, & (3.40) \end{aligned}$$

where $p_{\mathsf{W}}(w)$ is defined in Equation (3.24), and S_{W} denotes the *support set* of W, i.e. $S_{\mathsf{W}} = \{w : p_{\mathsf{W}}(w) > 0\}$.

The lower bound is given by

$$\hat{\mathrm{I}}_{\mathrm{lower}}(\mathsf{A};\mathsf{Y}) = \hat{\mathrm{H}}(\mathsf{Y}) - \hat{\mathrm{H}}(\mathsf{Y}|\mathsf{A}), \tag{3.41}$$

in which $\hat{\mathrm{H}}(\mathsf{Y}|\mathsf{A})$ is the sample mean of $-\log_2 p_{\mathsf{Y}|\mathsf{A}}(y_k|\mathbf{a}_k)$, for $k = 1 \ldots n$. It can be efficiently computed online using the following recursion (Rojas, 2003)

$$\begin{aligned} \mathrm{H}(\mathsf{Y}|\mathsf{A})_0 &= 0 \\ \mathrm{H}(\mathsf{Y}|\mathsf{A})_{k+1} &= \mathrm{H}(\mathsf{Y}|\mathsf{A})_k + \frac{1}{k+1}\left(-\log_2\left[p_{\mathsf{Y}|\mathsf{A}}(y_k|\mathbf{a}_k)\right] - \mathrm{H}(\mathsf{Y}|\mathsf{A})_k\right), \end{aligned} \tag{3.42}$$

where $\hat{\mathrm{H}}(\mathsf{Y}|\mathsf{A}) \equiv \mathrm{H}(\mathsf{Y}|\mathsf{A})_n$.

Both bounds can be proved by writing the difference $\mathrm{I}_{\mathrm{upper}}(\mathsf{A};\mathsf{Y}) - \mathrm{I}(\mathsf{A};\mathsf{Y})$, or respectively: $\mathrm{I}(\mathsf{A};\mathsf{Y}) - \mathrm{I}_{\mathrm{lower}}(\mathsf{A};\mathsf{Y})$ as a Kullback-Leibler divergence, which is always greater than or equal to zero. Details can be found in Arnold et al. (2002).

Examples on the dicode and exponential channel

Let us demonstrate the lower and upper bound on the binary-input *dicode* channel with Gaussian noise, which is a frequently-used channel model in magnetic recording. The D-transform of the channel impulse response is given by

$$h_{\text{dicode}}(D) = \frac{1-D}{\sqrt{2}}. \tag{3.43}$$

We have computed the upper and lower bound on the achievable information rate for the dicode channel in case of equiprobable binary signaling (signal alphabet: $\{-1,+1\}$), for $E_s/N_0 = -15\ldots15$ dB. We have used a blocklength of 10^6 samples per datum. Figure 3.3 shows the upper and lower bound, which approximately coincide.

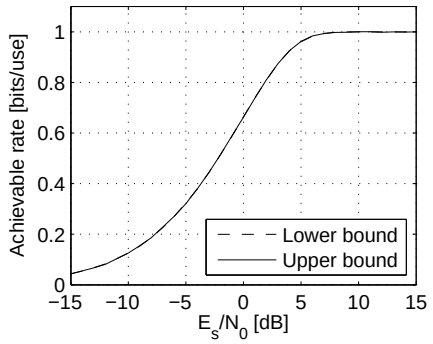
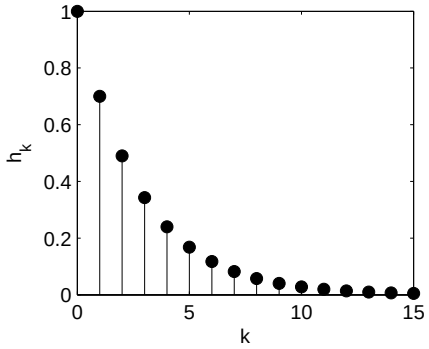


Figure 3.3: Achievable information rate on the binary-input dicode channel. Each datum was obtained using a blocklength of 10^6 samples. In this example, the upper and lower bound coincide. The plot is similar to that in Arnold and Loeliger (2001).

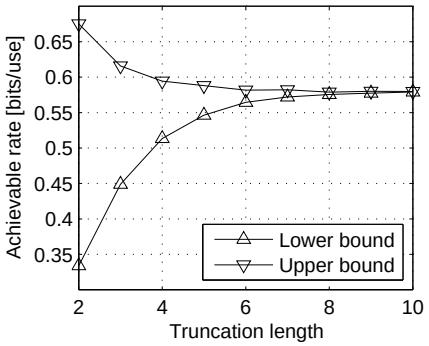
In the previous example, the dicode channel model was used to generate the channel output, and the same model was also used inside the BCJR algorithm. Let us now consider another example that more closely represents the intended application, i.e. in which the BCJR detector operates on a truncated version of the original channel. We introduce the exponential channel as

$$h_k = \alpha^k, \qquad k \in \{0,1,2,\ldots\}, \tag{3.44}$$

similar to Arnold et al. (2002). In this example, we assume $\alpha = 0.7$. Figure 3.4–a shows a stem plot of the first 16 coefficients of this impulse response. We compute the bounds on the achievable information rate for a scenario in which the noise has unit variance and the inputs are selected uniformly at random from the $\{-1,+1\}$-alphabet. For each simulation point, the channel output vector $\mathbf{y}$ is of length 10^6 and is generated using an impulse response of length 32. In Figure 3.4–b, the lower and upper bound are



(a) The first 16 coefficients of the discrete-time exponential channel impulse response, for $\alpha = 0.7$.



(b) Lower and upper bound on the achievable rate on the exponential channel, as a function of the truncation length of the impulse response used in the BCJR algorithm. Binary signaling, $\alpha = 0.7$, $\sigma^2_{\text{noise}} = 1$.

Figure 3.4: Plots belonging to the exponential channel example.

plotted, as a function of the truncation length of the impulse response that is used in the BCJR algorithm. As expected, the bounds become tighter when the truncation length is increased.

Chapter 4

Modulation and Coding for the Linear Gaussian Channel

In Chapter 3 we have considered the theoretical capacity of the linear Gaussian channel. In this chapter, we present a scheme with which this capacity can be approached in practice. We will first briefly discuss two existing papers on which our work is based. Subsequently, we explain the concept of multilevel coding and its mathematical basis. Finally, we present our proposed communication scheme, by distinguishing several subparts that we discuss in detail. To our knowledge, a major part of the work presented in Section 4.3 is new in the sense that several existing, but previously unconnected techniques are combined into a state-of-the-art communication scheme for the linear Gaussian channel.

4.1 Existing work

Soriaga et al. (2007) present a multilevel coding / multistage decoding system for binary intersymbol interference (ISI) channels, as encountered in magnetic recording. The binary ISI channel is an instance of the linear Gaussian channel, and hence, we can easily extend their work for our targeted scenario, i.e. spectral efficient communication using symbols from higher-order alphabets.

The second paper that we mention in this section is that of Cronie (2007)[1]. In this work, a modulation method is presented that generates a Gaussian-like input distribution by making linear combinations of binary input symbols. The method yields a shaping gain over for example ordinary

[1]Harm Cronie happens to be my supervisor in this project.

pulse amplitude modulation (PAM). In the paper, the method is demonstrated on the ideal (memoryless) AWGN channel at high SNR. We will apply the so-called *superposition modulation* technique to the linear Gaussian channel.

Both papers employ a form of multilevel coding. The following section is entirely devoted to this important concept.

4.2 Multilevel coding

The number of amplitude levels that should be used to optimally transmit information through a channel usually depends on the signal-to-noise ratio in that channel. For low signal-to-noise ratios, binary signaling suffices. However, for moderate to high SNRs, multiple (> 2) amplitude levels should be used. In this project, we focus on the latter scenario, in which binary signaling would be spectrally inefficient. However, the component codes that we use are *binary* LDPC codes, as motivated in Chapter 2. *Multilevel coding*, introduced by Imai and Hirakawa (1977), enables the use of (multiple) binary codes on a non-binary channel. With this approach, each binary code 'sees' its own independent binary channel.

Before going into details about multilevel coding, we want to point out that the word 'multilevel' is somewhat ambiguous. Although it is true that in our case multiple amplitude levels are used to communicate over a channel, the term 'multilevel' points to the abstract levels that appear when applying the *chain rule of mutual information*. In the original system proposed by Imai and Hirakawa (1977), the 'levels' correspond to the different bits of a symbol from a higher-order signal constellation, whereas in the paper of Soriaga et al. (2007), the interleaved binary codewords are called 'levels'. In our system, we consider levels in both dimensions, which we will call 'symbol levels' and 'time levels', respectively. We denote a certain level with two numbers placed inside a pair of superscripted square brackets, this is an extension of the notational style used in Richardson and Urbanke (2007) to two dimensions. In our notation, the first number denotes the symbol level, the second number represents the time level.

The multilevel concept is easiest explained using the original system of Imai and Hirakawa (1977). Consider a signal constellation S with M signal points. Let M be the Lth power of b, the base. We assume that $b = 2$. This means that every point in S can be addressed with a binary L-tuple. We transmit symbols of the constellation S sequentially over a channel. At the side of the receiver, an ordinary symbol detector would make one decision per transmitted signal point. Instead, we may use a *multistage* detector that makes L decisions per signal point, one for every address bit. The multistage decoder takes the previous decisions about address bits of a certain symbol into account.

As we mentioned before, multilevel coding enables the use of binary

codes on non-binary channels. Additionally, it provides fine-grained control over the placement of the redundancy of the error correcting codes. I.e., the key idea in multilevel coding is to 'protect' every address bit with a different code, having a rate that is tuned to the achievable rate in that subchannel.

4.2.1 Chain rule of mutual information

Mathematically, the multilevel concept is captured in the chain rule of mutual information, which states that the mutual information can be split in conditional parts. It is similar to the product rule of probability, which governs the factorization of a joint probability into conditional probabilities. For memoryless channels and a one-dimensional level structure, the chain rule for the mutual information between input X and output Y is written as

$$\mathrm{I}(X;Y) = \mathrm{I}(X^{[0]}, \cdots, X^{[L-1]}; Y) = \sum_{i=0}^{L-1} \mathrm{I}(X^{[i]}; Y | X^{[0]}, \cdots, X^{[i-1]}). \quad (4.1)$$

Because we focus on an ISI channel, we replace the scalars by vectors. Moreover, we switch to two-dimensional level indexing since we consider symbol and time levels separately. Let L_t be the number of time levels, and $L_s(t)$ the number of symbols levels in time level t. We then write:

$$\mathrm{I}(\mathbf{x};\mathbf{y}) = \sum_{t=0}^{L_t-1} \sum_{s=0}^{L_s(t)-1} \mathrm{I}(\mathbf{x}^{[s,t]}; \mathbf{y} | \mathbf{x}^{[0,0]}, \ldots, \mathbf{x}^{[s-1,t-1]}). \quad (4.2)$$

The conditional terms in Equation (4.2) represent the knowledge from previous decoding rounds. A drawback of such a dependency on prior information is the possibility of error propagation; if a codeword of a certain level is incorrectly decoded, all consequent levels are likely to be incorrectly estimated as well. However, the probability of such a catastrophic event will be very small if good codes with long enough blocklengths are used.

4.3 The proposed communication scheme

We present the proposed communication scheme by starting with an overview of the entire system, in which we distinguish several subblocks. Subsequently, we deal with the individual subblocks in detail.

4.3.1 System overview

Figure 4.1 shows an overview of the communication scheme. The information source is assumed to output uniformly distributed bits, i.e. $p(0) = p(1) = 1/2$. As can be seen from the figure, the actual system (everything except the source and sink) has been divided into three parts. We start with

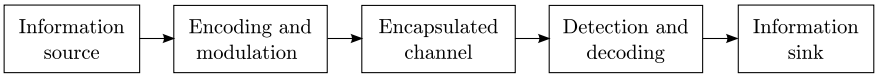


Figure 4.1: Overview of the communication scheme. The 'encapsulated channel' represents the physical channel together with the required analog prefiltering and detection operations.

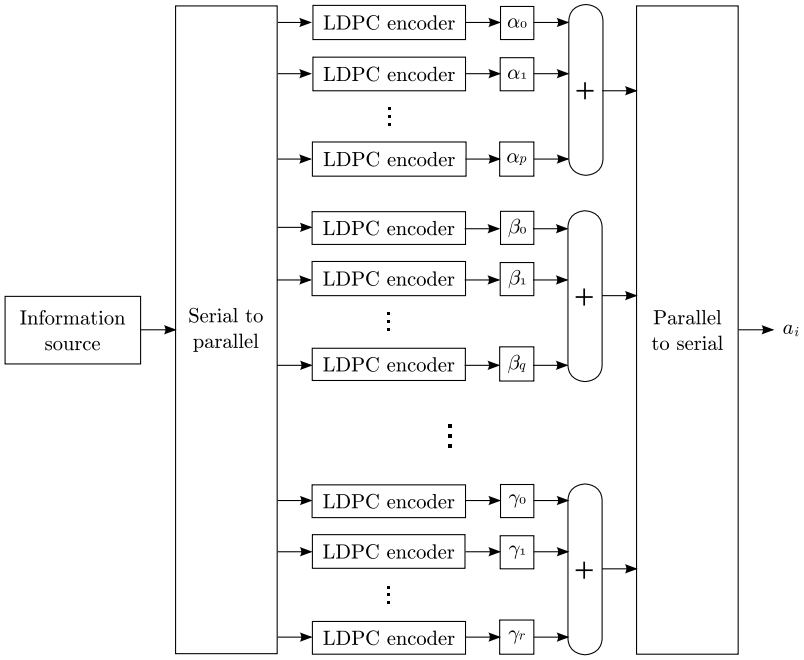


Figure 4.2: Structure of the encoding and modulation process at the transmitter.

an analysis of the encoding and modulation operations that take place at the side of the transmitter.

4.3.2 Encoding and modulation

Figure 4.2 shows the structure of the encoder/modulator. We discuss details about the various blocks in separate subsections.

Serial to parallel conversion and LDPC encoding

Every component code has its own rate[2], but they all share a common codeword length. Consequently, the number of information bits that is mapped on a codeword is different in each level. The serial-to-parallel (S2P) block is responsible for distributing the bits from the information source over

[2]A method to determine these rates will be discussed in Section 4.3.5.

the input buffers of the encoders, at rates equal to the component code rates. This ensures that the outputs of the encoders run in pace.

To simulate the system, we do not need to implement the S2P block and the LDPC encoders when we make use of coset randomization. For details refer to Chapter 2.

Superposition modulation

As shown in Figure 4.2, the outputs of the LDPC encoders (which emit bits chosen from $\{-1, +1\}$) are connected to adders via scalar multipliers. The additions and multiplications take place in $\mathbb{R}$, not in $GF(2)$. The multiplier coefficients, in the figure denoted as α_i, β_i and γ_i, determine the signal constellations. Imai and Hirakawa (1977) set those coefficients to powers of two, yielding the well-known M-PAM constellations. Cronie (2007) introduced new signal constellations by altering the coefficients. The key advantage of these alternative constellations is that the amplitude distribution of the output becomes Gaussian-like (by the central limit theorem), yielding a shaping gain on additive Gaussian noise channels.

In the paper of Cronie (2007), two classes of signal constellations are introduced. *Binomial constellations* are obtained when all the coefficients are set to one. The binomial constellation works very well with just a few bits. However, it does not scale well to higher spectral efficiencies, because the entropy of the binomial distribution grows only logarithmically in the number of bits. Therefore, a second constellation type is introduced that performs better in the high-SNR regime. The coefficients of this constellation type are found using non-linear numerical optimization. Hence, the family of constellations is called *numerically optimized constellations*. For low signal-to-noise ratios, and with that low spectral efficiencies, numerically optimized constellations tend to converge to binomial constellations.

Interleaving

In Figure 4.2, we distinguish several groups consisting of LDPC encoders, scalar multipliers and one adder. Each of these groups corresponds to a time level. The parallel-to-serial (P2S) block interleaves the adder outputs. The interleaving depth is equal to the number of time levels.[3] The number of encoder-multiplier pairs in a group corresponds to the number of symbols levels in the associated time level.

Let us clarify this by an example. Assume a system with two time levels. The time levels contain two and three symbol levels, respectively. Let $c_k^{[s,t]}$ denote the kth bit from an LDPC codeword in the sth symbol level and the

[3] We define the interleaving depth such that a depth of one represents no interleaving.

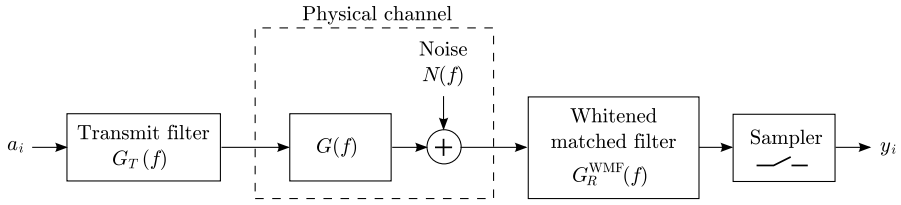


Figure 4.3: Detailed view of the 'encapsulated channel'.

tth time level. The output a_i is constructed as follows:

$$\begin{aligned} a_0 &= \alpha_0 c_0^{[0,0]} + \alpha_1 c_0^{[1,0]} \\ a_1 &= \beta_0 c_0^{[0,1]} + \beta_1 c_0^{[1,1]} + \beta_2 c_0^{[2,1]} \\ a_2 &= \alpha_0 c_1^{[0,0]} + \alpha_1 c_1^{[1,0]} \\ a_3 &= \beta_0 c_1^{[0,1]} + \beta_1 c_1^{[1,1]} + \beta_2 c_1^{[2,1]} \\ \ldots &= \ldots \end{aligned}$$

Note that the odd and even samples of a_i are chosen from different signal constellations. ■

Strictly speaking, bits from LDPC codewords are not entirely uncorrelated. Nevertheless, we assume that the output a_i has approximately a flat power spectral density.

4.3.3 The channel

With 'channel' we refer to the equivalent discrete-time channel that encapsulates the transmit filter (creating a continuous-time output signal with the optimal water-pouring power spectral density), the physical analog channel (modeled as a linear filter followed by a Gaussian noise source) and the whitened matched filter and sampler at the receiver, as illustrated in Figure 4.3.

The transmit filter assumes that the input process a_i has a flat power spectral density (as produced by the encoder/modulator), such that

$$\frac{1}{T} P_a(f) \; |G_T(f)|^2 = \frac{\sigma_a^2}{T} |G_T(f)|^2 = P_s^o(f). \tag{4.3}$$

(The definition of the symbols that appear in this and the following equation can be found in Chapter 3.) This filtering operation at the transmitter yields some additional shaping gain; it makes the probability distribution of the amplitude of the signal more Gaussian-like, because of the central limit theorem.

In Figure 4.3, $G_R^{\mathrm{WMF}}(f)$ represents the composite receive filter consisting of the noise-whitening filter and the whitened matched filter. Its transfer

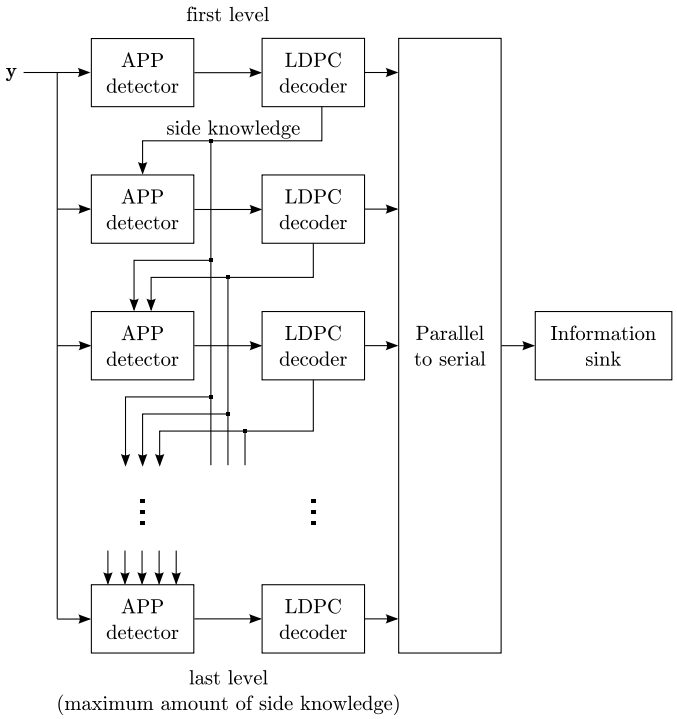


Figure 4.4: The multistage detector/decoder.

function is given by (Forney and Ungerboeck, 1998)

$$G_R^{\mathrm{WMF}}(f) = \frac{V^*(f)}{A^2\tilde{H}^*(f)} = \frac{G_T^*(f)G^*(f)\tilde{H}(f)}{N(f)\tilde{Q}(f)}. \tag{4.4}$$

Details about the stability conditions of this filter can be found in the provided reference.

4.3.4 Multistage detection and decoding

The multistage detector/decoder is depicted in Figure 4.4. Every stage consists of an *a posteriori* probability (APP) detector followed by an LDPC decoder. The estimated codeword of the first level is used as side knowledge by the APP detector for the second level. The APP detector of the third level uses side knowledge from the first and second LDPC decoder. This decoding process is repeated until the last level is reached, in which the APP detector benefits from all previous decodings. This particular way of handling the side information is also shown in Figure 4.4 by the network of arrows between the blocks.

The APP detector is based on the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm, which is a synonym for the sum-product algorithm that is applied

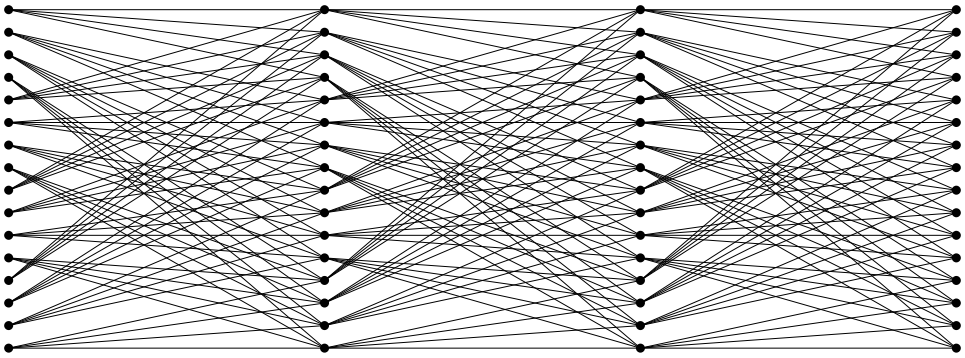


Figure 4.5: Three concatenated trellis sections of a multilevel communication system having two time levels and two symbol levels per time level (two bits per symbol). Memory length: $M = 2$.

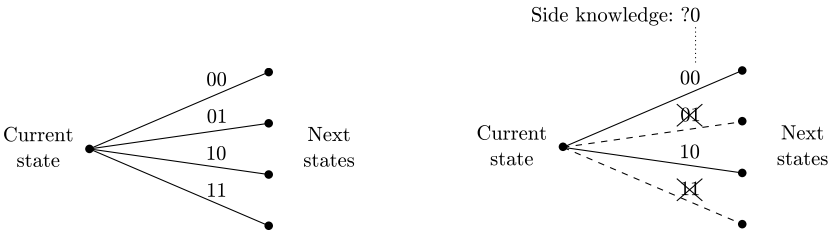
to a *trellis*. In Figure 4.5 we see an example of a trellis. The illustration depicts three concatenated trellis sections, for an example system having two time levels, each consisting of two symbol levels, and a channel memory length equal to two. The BCJR algorithm is also known as the 'belief-propagation' and 'forward-backward' algorithm. The algorithm computes the *a posteriori* probability for each state in a trellis, using a forward and a backward recursion. For details about the algorithm, we refer to the original publication by Bahl et al. (1974), or for instance to Forney's lecture notes and video lectures at the MIT OpenCourseWare website.

From Figure 4.4 we see that code bits that belong to the first level are detected without prior knowledge. In the subsequent levels, the available side information is taken into account by the BCJR algorithm in a way that is depicted in Figure 4.6. The figure shows that side knowledge decreases the number of possible transitions in a trellis section. As more side knowledge becomes available, the BCJR algorithm has to run through less states and therefore becomes faster.

When the allocation of symbol levels over the time levels is non-uniform, the number of trellis states is likely to vary with the time index. An graphical example of such a situation is shown in Figure 4.7. Let us explain this effect by another example. Assume that we have the same system as in the example in Section 4.3.2, i.e. two time levels, with two and three symbol levels, respectively. Furthermore assume that a four-tap discrete-time channel approximation is used, hence the memory length $M = 3$. We label the five binary subchannels with a single integer x, which is a function of the symbol level s and the time level t:

$$x = s + \sum_{\tau=0}^{t-1} L_s(\tau). \tag{4.5}$$

We introduce a tabular representation, in which the horizontal direction



(a) Example of a two-bit transition (four possible transitions).

(b) According to the side knowledge the second bit equals zero, which makes two transitions invalid.

Figure 4.6: Incorporation of side knowledge in the BCJR algorithm. The knowledge reduces the number of possible transitions in a trellis section, thereby increasing the probability mass of the remaining transitions.

corresponds to the time axis and the vertical direction corresponds to the symbol levels. In conjunction with the labeling from Equation (4.5), we can express an arbitrary piece of the output of the modulator as:

$$\begin{array}{ccccccccccccc} \ldots & 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 & 0 & 2 & \ldots \\ & 1 & 3 & 1 & 3 & 1 & 3 & 1 & 3 & 1 & 3 & 1 & 3 & \\ & & 4 & & 4 & & 4 & & 4 & & 4 & & 4 & \end{array}$$

We can also express the cardinality of the time levels (in other words: the number of symbol levels in each time level) in the following fashion:

$$\ldots \quad 2 \quad 3 \quad 2 \quad 3 \quad 2 \quad 3 \quad 2 \quad 3 \quad 2 \quad 3 \quad 2 \quad 3 \quad \ldots$$

In the forward pass, the BCJR algorithm runs over this sequence from left to right, using $M+1$ adjacent symbols from this sequence per trellis section, as indicated below:

$$\ldots \quad 2 \quad 3 \quad \boxed{2 \quad 3 \quad 2 \quad 3} \quad 2 \quad 3 \quad 2 \quad 3 \quad 2 \quad 3 \quad \ldots$$

M past samples current sample

The M past samples form the *state*. In this case, the number of states N equals

$$N = 2^{(2+3+2)} = 128. \tag{4.6}$$

In the next iteration (the next trellis section), we have the following situation:

$$\ldots \quad 2 \quad 3 \quad 2 \quad \boxed{3 \quad 2 \quad 3 \quad 2} \quad 3 \quad 2 \quad 3 \quad 2 \quad 3 \quad \ldots$$

The state is now made up of eight bits (instead of seven), corresponding to $2^8 = 256$ possible states. This shows that the number of states in a trellis section can vary with the time index. ■

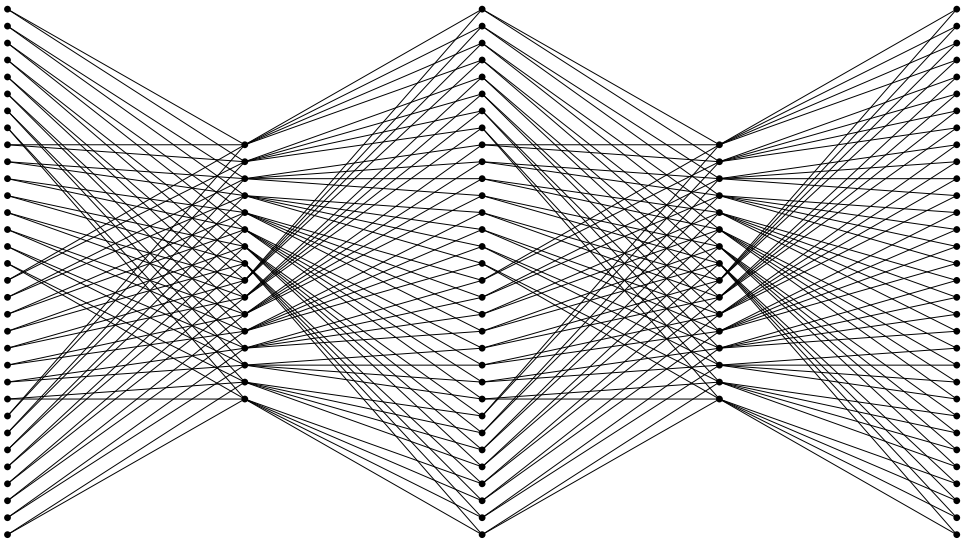


Figure 4.7: An example of a trellis in which the number of trellis states is not constant, but varies in a deterministic way. This happens for particular combinations of the non-uniform level allocation and the value for the memory length (M). This example is based on a scheme with two time levels, one symbol level in the first time level (binary signaling), two symbol levels in the second time level, and a memory length equal to three.

From APP vectors to probabilities and to log likelihood ratios

Following its name, the APP detector from a certain level should output probabilities for the bits that correspond to that level. The BCJR algorithm computes state probabilities, for all time levels, and outputs APP vectors. The BCJR output can be converted to the desired probabilities by means of three steps. The first step is to keep only the APP vectors from the time indices that belong to the level that we are interested in. The second step is to partition the elements of these remaining APP vectors into two disjunct sets, based on the value of the bit that corresponds to the symbol index of the level of interest. Because of this partitioning strategy, the resulting sets will not only be disjunct, but also equal in size. The third step is to sum the APP values of one of the two sets, depending on whether we want to compute the probability of a '0' or a '1'. It does not really matter which probability – $p(0)$ or $p(1)$ – we compute, since their sum equals one, e.g. $p(1) = 1 - p(0)$. Using this property, the log likelihood ratio ℓ is written as

$$\ell = \log \frac{p(0)}{1 - p(0)}. \tag{4.7}$$

The log likelihood ratios are fed to the LDPC decoders, and are sometimes referred to as the intrinsic information.

4.3.5 Achievable subchannel rates

We estimate the achievable subchannel rates using a Monte Carlo simulation, following Soriaga et al. (2007). The basic idea is to simulate a huge amount of channel output, run the BCJR algorithm on it, and compute the achievable subchannel rates from the collected marginal conditional output densities. In their paper, we find the following expression

$$R = \lim_{N\to\infty} \frac{1}{N} \sum_{j=1}^{N} \left[1 - \log_2\left(1 + \exp\left(-c_j \ell_j\right)\right)\right], \qquad c_j \in \{\pm 1\} \tag{4.8}$$

where c_j is the jth code bit (or the coset bit if the all-one codeword is used) and ℓ_j the log likelihood ratio for the jth bit. The expression originates from Chung (2000). Actually, we do not even need to compute ℓ_j. Using

$$\ell_j = \log \frac{p_j}{1 - p_j}, \tag{4.9}$$

we write Equation (4.8) as

$$\begin{aligned} R &= \lim_{N\to\infty} \frac{1}{N} \sum_{j=1}^{N} \left[1 - \log_2\left(1 + \left(\frac{1-p_j}{p_j}\right)^{c_j}\right)\right] \\ &= \lim_{N\to\infty} \frac{1}{N} \sum_{j=1}^{N} \left[1 + \log_2\left(c_j\left(p_j - \frac{1}{2}\right) + \frac{1}{2}\right)\right]. \end{aligned} \tag{4.10}$$

The order in which the levels are detected and decoded can be chosen arbitrarily. Although the decoding order will influence the allocation of the subchannel rates, the overall (cumulative) rate will remain constant.

Chapter 5

Simulations and Results

In this chapter, we demonstrate most of the techniques that we have discussed in the preceding chapters on a simple channel model. We consider an RC low-pass filter circuit, simulating a wireline channel. It has a frequency-selective response, and together with an additional white noise source, it is a typical instance of the linear Gaussian channel. Moreover, the RC wireline channel has an amplitude-continuous input, that we can exploit by using superposition modulation. First, the capacity of this channel is computed using the expressions from Chapter 3. Then, we will derive a discrete-time finite-length channel representation, and use this to compute the bounds on the achievable information rate, for various signal constellations. Moreover, we determine achievable subchannel rates, for multiple setups. For a particular setup, we design and optimize LDPC component codes and benchmark the system using bit-error rate (BER) simulations.

Please note that in this chapter, R and C represent 'resistance' and 'capacitance', respectively, instead of the information-theoretic quantities 'rate' and 'capacity'.

5.1 The RC wireline channel

The behavior of the RC low-pass network (shown in Figure 5.1) as a function of frequency is described by its transfer function

$$G(f) = \frac{1}{1 + \mathrm{j}2\pi f RC}. \tag{5.1}$$

For the major part of the analysis only the product of R and C matters, hence we use the time constant τ which is defined as

$$\tau = RC. \tag{5.2}$$

The magnitude response (being the squared modulus of the transfer function) is equal to

$$|G(f)|^2 = \frac{1}{1 + 4\pi^2\tau^2 f^2}. \tag{5.3}$$

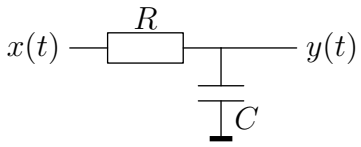


Figure 5.1: The RC low-pass filter circuit, that is used as channel model.

The 'cut-off frequency', the '-3 dB point' or simply 'the frequency where the (two-sided) magnitude response is equal to one-half' lies at

$$f_{\text{cut}-\text{off}} = \frac{1}{2\pi\tau}. \tag{5.4}$$

We assume white noise, having the two-sided power spectral density

$$N(f) = \frac{N_0}{2}. \tag{5.5}$$

We assume a scenario with an average power constraint, but without a bandwidth constraint. We furthermore assume that the capacity-achieving band is a single frequency interval. The width of this band depends on K, refer to Equation (3.11). To find this width, the water-pouring solution (Equation (3.10)) $K - 1/\text{SNR}_\text{c}(f)$ is first equated to zero:

$$\begin{aligned} K - \frac{1}{\text{SNR}_\text{c}(f)} &= \\ K - \frac{N_0}{2}(1 + 4\pi^2\tau^2 f^2) &= 0. \end{aligned} \tag{5.6}$$

Subsequently, this equation is solved for f under the constraint that $f > 0$

$$f_{B,\text{edge}} = \frac{\sqrt{N_0(2K - N_0)}}{2N_0\pi\tau}. \tag{5.7}$$

From this result, we see that $2K \geq N_0$ in order to obtain a valid (real-numbered) frequency.

Now that we have found an expression for the edge of the capacity-achieving band, we can calculate the transmit power as a function of K

$$\begin{aligned} P &= \int_{-f_{B,\text{edge}}}^{f_{B,\text{edge}}} P_s^o(f)\mathrm{d}f \\ &= \int_{-f_{B,\text{edge}}}^{f_{B,\text{edge}}} \left[K - \frac{N_0}{2}(1 + 4\pi^2\tau^2 f^2)\right] \mathrm{d}f \\ &= \frac{(2K - N_0)^{\frac{3}{2}}}{3\pi\tau\sqrt{N_0}} \text{ [W]}. \end{aligned} \tag{5.8}$$

In this form, the expression is not useful, we would rather have an expression for K in terms of P. Therefore, we take Equation (5.8) and isolate K under the (valid) assumption that K is real, yielding

$$K = \frac{N_0}{2} + \frac{1}{2}\left(3\pi P\tau\right)^{\frac{2}{3}} \sqrt[3]{N_0}. \tag{5.9}$$

The expression for K is substituted into Equation (5.7), resulting in

$$f_{B,\text{edge}} = \frac{\sqrt[3]{3P}}{2\sqrt[3]{N_0}\left(\pi\tau\right)^{\frac{2}{3}}}. \tag{5.10}$$

Next, we express P in terms of the more common symbol energy E_s, so that we can work with the familiar signal-to-noise ratio

$$\frac{E_s}{N_0}. \tag{5.11}$$

P and E_s are related via the symbol period T:

$$P = \frac{E_s}{T}. \tag{5.12}$$

T equals $1/(2W)$ because of minimum-bandwidth transmission of real-valued symbols. The (one-sided) bandwidth W is equal to $f_{B,\text{edge}}$. Together with Equation (5.12), this results in

$$P = 2E_s f_{B,\text{edge}}. \tag{5.13}$$

Because $f_{B,\text{edge}}$ is expressed in terms of P, we manipulate Equation (5.13) and bring all instances of P to the left hand side:

$$\begin{aligned} P &= 2E_s \frac{\sqrt[3]{3P}}{2\sqrt[3]{N_0}\left(\pi\tau\right)^{\frac{2}{3}}} \\ &= \frac{{E_s}^{\frac{3}{2}}\sqrt{3}}{\pi\tau\sqrt{N_0}}. \end{aligned} \tag{5.14}$$

For the sake of completeness, we also write $f_{B,\text{edge}}$ and K as a function of E_s instead of P

$$f_{B,\text{edge}} = \frac{\sqrt{3}}{2\pi\tau}\sqrt{\frac{E_s}{N_0}}, \tag{5.15}$$

$$K = \frac{1}{2}N_0 + \frac{3}{2}E_s. \tag{5.16}$$

Note the remarkably simple relation between N_0, E_s and K.

5.1.1 Capacity and spectral efficiency

The capacity as a function of E_s, N_0 and τ is given by

$$\begin{aligned} C &= \int_0^{f_{B,\text{edge}}} \log_2 \left[\frac{2K|G(f)|^2}{N_0} \right] \mathrm{d}f \\ &= \int_0^{\frac{\sqrt{3E_s}}{2\pi\tau\sqrt{N_0}}} \log_2 \left[\frac{N_0 + 3E_s}{N_0(1+4\tau^2\pi^2 f^2)} \right] \mathrm{d}f \\ &= \frac{2E_s\sqrt{3N_0} + 2N_0\sqrt{E_s}\arctan\left(\frac{\sqrt{3N_0}}{3\sqrt{E_s}}\right) - \pi N_0\sqrt{E_s}}{2\log(2)\pi\tau N_0\sqrt{Es}} \\ &= \frac{\sqrt{3}\sqrt{\frac{E_s}{N_0}} - \arctan\left(\sqrt{3}\sqrt{\frac{E_s}{N_0}}\right)}{\log(2)\pi\tau} \text{ [bits/s].} \end{aligned} \tag{5.17}$$

The latter expression was derived using the following identities:

$$\int \log\left(x^2+a^2\right)\mathrm{d}x = x\log\left(x^2+a^2\right) - 2x + 2a\arctan\left(\frac{x}{a}\right), \tag{5.18}$$

$$\arctan\left(\frac{1}{x}\right) = \frac{\pi}{2} - \arctan(x), \qquad x > 0. \tag{5.19}$$

The spectral efficiency (specified in bits per dimension), is obtained as follows

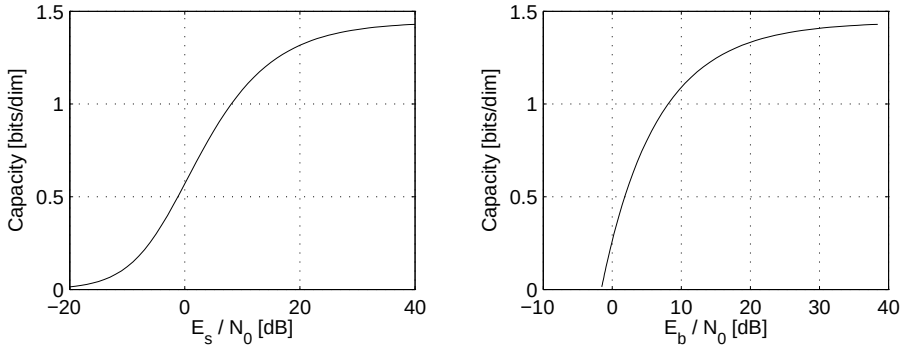
$$\begin{aligned} \frac{C}{D}\left(\frac{E_s}{N_0}\right) &= \frac{C}{2f_{B,\text{edge}}} \\ &= \frac{\sqrt{3}\sqrt{\frac{E_s}{N_0}} - \arctan\left(\sqrt{3}\sqrt{\frac{E_s}{N_0}}\right)}{\sqrt{3}\sqrt{\frac{E_s}{N_0}}\log(2)} \text{ [bits/dim].} \end{aligned} \tag{5.20}$$

The spectral efficiency is plotted as a function of E_s/N_0 and E_b/N_0 in Figure 5.2. The figure shows that C/D approaches a limit for infinite SNR

$$\lim_{E_s/N_0\to\infty} \frac{C}{D}\left(\frac{E_s}{N_0}\right) = \frac{1}{\log 2} \approx 1.44 \text{ [bits/dim].} \tag{5.21}$$

5.1.2 Derivation of the canonical discrete-time model

In order to find the coefficients of the equivalent canonical discrete-time model, we start with the end-to-end response $Q(f)$, which was defined in Equation (3.16). Without loss of generality, we assume that the channel



(a) As a function of E_s/N_0. (b) As a function of E_b/N_0.

Figure 5.2: The spectral efficiency of the RC channel.

input sequence a_i has unit variance: $\sigma_a^2 = 1$.

$$\begin{aligned} Q(f) &= \frac{T}{\sigma_a^2} P_s^o(f) \mathrm{SNR_c}(f) \\ &= T\left(\frac{2K|G(f)|^2}{N_0} - 1\right) \\ &= T\left(\frac{N_0 + 3E_s}{N_0(1 + 4\pi^2\tau^2 f^2)} - 1\right). \end{aligned} \tag{5.22}$$

In the band $[-f_{B,\mathrm{edge}}, f_{B,\mathrm{edge}}]$, $\tilde{Q}(f)$ is equal to $Q(f)/T$. We subsitute this into Equation (3.20) (introduced in Section 3.3.1), yielding the following expression:

$$\begin{aligned} \alpha_\ell &= T\int_{\langle 1/T\rangle} \log\left[\tilde{Q}(f)\right] \exp(\mathrm{j}2\pi f\ell T)\mathrm{d}f \\ &= T\int_{-f_{B,\mathrm{edge}}}^{f_{B,\mathrm{edge}}} \log\left[\frac{N_0 + 3E_s}{N_0(1 + 4\pi^2\tau^2 f^2)} - 1\right] e^{\mathrm{j}2\pi f\ell T}\mathrm{d}f, \end{aligned} \tag{5.23}$$

where $T = 1/(2f_{B,\mathrm{edge}})$. Using $\phi = \tau f$, we write

$$\alpha_\ell = \frac{\pi}{\sqrt{3}\sqrt{\dfrac{E_s}{N_0}}} \int_{-\frac{\sqrt{3}}{2\pi}\sqrt{\frac{E_s}{N_0}}}^{\frac{\sqrt{3}}{2\pi}\sqrt{\frac{E_s}{N_0}}} \log\left[\frac{N_0 + 3E_s}{N_0(1 + 4\pi^2\phi^2)} - 1\right] \exp\left(\frac{\mathrm{j}2\pi^2\phi\ell\sqrt{N_0}}{\sqrt{3E_s}}\right)\mathrm{d}\phi. \tag{5.24}$$

In words, this means that α_ℓ is independent of τ, and implies that a discrete-equivalent model for a first-order RC network will hold for any τ. The SNR is the only design parameter. Unfortunately, the integral in Equation (5.24) cannot be evaluated in closed form, which means that our analytic analysis of the discrete-time model ends here. By the way, Equation (5.24) can be computed numerically without problems.

E_s/N_0 [dB]	C/D [bits/dim]	Truncated impulse response	A^2
7.0	0.95217	$[1, 1.0677, 0.3854, 0.2170]$	1.9001

Table 5.1: The canonical discrete-time model for $E_s/N_0 = 7$ dB.

As mentioned before, the BCJR algorithm should be run on a truncated version of the discrete-time impulse response $\{h_k\}$. The truncated impulse response can be regarded as an approximation of the infinite-tap discrete-time model. The better this approximation, the tighter the bounds on the information rate will become and the closer the subchannel rates will approach their maximums. From numerical experiments, it is found that the number of coefficients needed to get an accurate approximation of $\tilde{Q}(f)$ increases with the SNR. In other words, for a fixed number of filter coefficients, the approximation deteriorates as the SNR increases. In the next section, we will demonstrate the theory by means of a design example. Because of the effect that we have just discussed, we should target at a scenario in which the SNR is low enough to obtain a fairly accurate discrete-time approximation with just a few (say, four) filter coefficients. On the other hand, the SNR should be high enough to justify the use of a higher-order signal constellation and with that superposition coding.

5.2 A design example

We now demonstrate how to design a spectral-efficient communication system for the first-order RC channel with AWGN. Assume a scenario in which $E_s/N_0 = 7$ dB. According to Equation (5.20) and Figure 5.2–a, this corresponds to a maximum spectral efficiency of 0.95217 bits/dim. Since this is a moderate[1] spectral efficiency, we will attempt to achieve this capacity with a binomial signal constellation.

The first step in the design process is to derive a canonical discrete-time model. The first ten coefficients of the resulting discrete-time impulse response are shown in Figure 5.3. For the APP detector (BCJR algorithm), we will use a truncated version of the impulse response having only four coefficients. The values of these coefficients can be found in Table 5.1. To generate the very long channel output sequence, we use a 32-tap version of the canonical discrete-time model.

The next step is to determine a reasonable *level allocation* for our system, by which we mean the number of time levels and the number of symbol levels in each time level. For the sake of simplicity, we start by analyzing level allocations in which the number of symbol levels is the same in all time levels. We call these *uniform* level allocations.

[1] 'moderate' as in Cronie (2007).

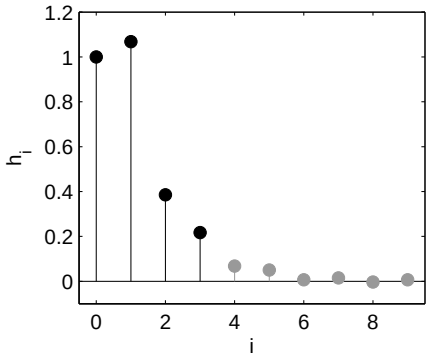
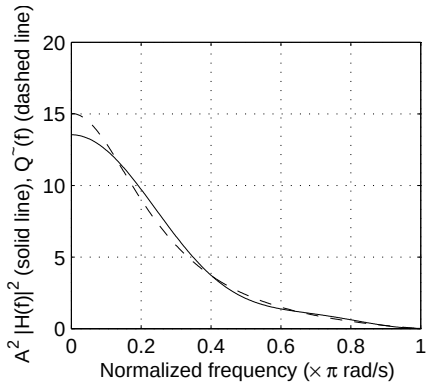
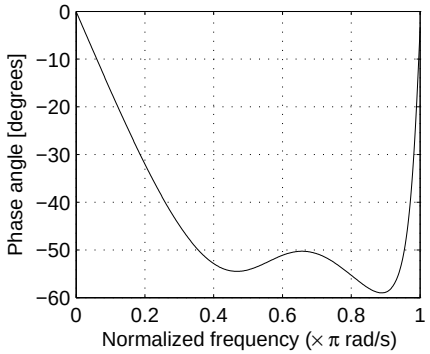


Figure 5.3: The first ten coefficients of the discrete-time canonical channel model, derived for $E_s/N_0 = 7$ dB. As a trade-off between accuracy and computational complexity, only the first four coefficients will be used in the APP detector / BCJR algorithm.



(a) Comparison between the desired response $\tilde{Q}(f)$ (dashed line) and the actual response $A^2|H(f)|^2$.



(b) The phase response.

Figure 5.4: Magnitude and phase response of the discrete-time four-coefficient model ($E_s/N_0 = 7$ dB).

Bits per symbol	Constellation type	Unnormalized signal set	Estimated information rate [bits/dim]	Rate-to-capacity ratio
1	Binary	$\{\pm 1\}$	$0.8496 \ldots 0.8591$	0.897
2	Binomial	$\{0, \pm 2\}$	$0.9339 \ldots 0.9439$	0.986
2	4-PAM	$\{\pm 1, \pm 3\}$	$0.9245 \ldots 0.9354$	0.977
3	Binomial	$\{\pm 1, \pm 3\}$	$0.9387 \ldots 0.9498$	0.992
3	8-PAM	$\{\pm 1, \pm 3, \pm 5, \pm 7\}$	$0.9295 \ldots 0.9405$	0.982

Table 5.2: Information rates, $E_s/N_0 = 7$ dB. The information rates are specified as intervals between the lower and upper bound. The rate-to-capacity ratio is computed as the average of the lower and upper bound, divided by the capacity (0.95217 bits/dim).

For various numbers of symbol levels (the number of symbol levels is equivalent to the number of bits per symbol), we compute the lower and upper bound for the achievable information rate. (These bounds and the accompanying Monte Carlo algorithm are discussed in Section 3.5.) This 'achievable information rate' is approached when the number of time levels goes to infinity (Soriaga et al., 2007). We not only simulate using binomial signal constellations, but also using PAM constellations, to check whether the binomial signal constellation does indeed achieve shaping gain. The results of these simulations, for which we used a blocklength of 10^6 samples, can be found in Table 5.2. From these results, we see that by going from one to two bits per symbol, the achievable rate increases by ten percent. Adding more bits per symbol only gives a marginal gain in performance. The shaping gain achieved by superposition modulation over ordinary PAM amounts to roughly 1%, which is actually not bad for the moderate SNR ($E_s/N_0 = 7$ dB) that we consider. An additional interesting result is that the two-bit binomial signal constellation even slightly outperforms the (three-bit) 8-PAM constellation.

We have computed estimates for the achievable binary subchannel rates, using binary and two- and three-bit binomial constellations and a varying number of time levels (one to four). The results for the binary constellation are printed in Table 5.3. For binary signaling, the algorithm to determine the subchannel rates is equal to the method described in Soriaga et al. (2007). Achievable subchannel rates for higher-order binomial constellations are shown in Table 5.4 and Table 5.5. In the latter two tables, note the legend in the lower right. Moreover, note the significant difference between the average rates corresponding to one and two time levels. For these simulations, we used a blocklength of 10^5 samples to limit the memory usage and the duration of the simulation.

The $(0, 0)$ table entries represent the rate of the firstly decoded subchannel, for which no prior information is available. Consequently, the value of

	0	1
Subch. rate	0.7610	0.9197
Avg.	0.8403	

(a) Two time levels.

	0	1	2
Subch. rate	0.7580	0.8565	0.9259
Avg.	0.8468		

(b) Three time levels.

	0	1	2	3
Subch. rate	0.7628	0.8451	0.8604	0.9267
Avg.	0.8488			

(c) Four time levels.

Table 5.3: Achievable subchannel rates with binary signaling, for a varying number of time levels. $E_s/N_0 = 7$ dB, blocklength: 10^5 samples.

this rate depends on the number of symbol levels (the constellation type) in the first time level, but it is independent of the number of subsequent time levels. I.e. the top-left entries from tables that show results obtained from the same constellation should ideally be equal. In practice, the estimated rates will of course not be exactly equal because we can only simulate with finite blocklengths, but we can use this fact to get an idea of the accuracy of the estimates. Regarding Table 5.3 to 5.5, we can infer that the results are accurate up to roughly two decimals.

5.2.1 Non-uniform allocation of symbol levels

So far, we have used a uniform number of symbol levels per time level. In the lower time levels, the effective signal-to-noise ratio is low because of the uncancelled intersymbol interference. Therefore, in these lower time levels, we can try to spare symbol levels (i.e. use less bits per symbol). An advantage of reducing the number of levels is that the complexity of the BCJR algorithm decreases, since it has to run through less states. (However, implementing a BCJR algorithm in which the number of states is no longer fixed but dependent on the time level becomes a bit of a hassle.) Table 5.6 shows achievable overall rates, obtained by using various non-uniform level allocations. We denote a level allocation as a tuple of length L_t (L_t is the number of time levels). The ith tuple element contains the number of symbol levels in the it time level. Table 5.7 shows achievable subchannel rates for various setups.

Comparing these results to those from Table 5.4 and Table 5.5, we can conclude that under these circumstances the non-uniform allocation of symbol levels over time levels does not bring us something spectacular. We expect that non-uniform level allocation will be of more importance at higher spectral efficiencies. However, as the number of required symbol levels in-

	0
0	0.2808
1	0.5300
Cum.	0.8108
Avg.	0.8108

(a) One time level.

	0	1
0	0.2809	0.3847
1	0.4509	0.7218
Cum.	0.7318	1.1065
Avg.	0.9192	

(b) Two time levels.

	0	1	2
0	0.2747	0.3348	0.3987
1	0.4469	0.5911	0.7452
Cum.	0.7216	0.9259	1.1438
Avg.		0.9304	

(c) Three time levels.

	0	1	2	3
0	0.2785	0.3310	0.3459	0.3995
1	0.4459	0.5796	0.6050	0.7455
Cum.	0.7244	0.9106	0.9509	1.1450
Avg.		0.9327		

(d) Four time levels.

	Time level
Symbol level	Subchannel rates
Cum.	Column-wise sums
Avg.	Average rate

(e) Table legend.

Table 5.4: Achievable rates for the binary subchannels, obtained using a binomial signal constellation with two bits per symbol, for a varying number of time levels. $E_s/N_0 = 7$ dB, blocklength: 10^5 samples.

	0
0	0.1681
1	0.2369
2	0.4235
Cum.	0.8284
Avg.	0.8284

(a) One time level.

	0	1
0	0.1670	0.2292
1	0.2225	0.3283
2	0.3229	0.5805
Cum.	0.7124	1.1380
Avg.	0.9252	

(b) Two time levels.

	0	1	2
0	0.1685	0.1993	0.2332
1	0.2179	0.2802	0.3398
2	0.3217	0.4466	0.6037
Cum.	0.7081	0.9261	1.1762
Avg.		0.9370	

(c) Three time levels.

	0	1	2	3
0	0.1693	0.2016	0.2063	0.2344
1	0.2200	0.2762	0.2859	0.3450
2	0.3166	0.4374	0.4564	0.6105
Cum.	0.7059	0.9152	0.9485	1.1899
Avg.		0.9399		

(d) Four time levels.

	Time level
Symbol level	Subchannel rates
Cum.	Column-wise sums
Avg.	Average rate

(e) Table legend.

Table 5.5: Achievable rates for the binary subchannels, obtained using a binomial signal constellation with three bits per symbol, for a varying number of time levels. $E_s/N_0 = 7$ dB, blocklength: 10^5 samples.

Level allocation	Estimated information rate [bits/dim]	Rate-to-capacity ratio
(1, 2)	0.9039 . . . 0.9149	0.955
(2, 3)	0.9341 . . . 0.9473	0.988
(1, 2, 3)	0.9214 . . . 0.9337	0.974
(1, 2, 3, 4)	0.9247 . . . 0.9382	0.978

Table 5.6: Achievable overall information rates for the various non-uniform level allocations. All constellations are binomial. $E_s/N_0 = 7$ dB. The information rates are specified as intervals between the lower and upper bound. The rate-to-capacity ratio is computed as the average of the lower and upper bound, divided by the capacity (0.95217 bits/dim).

	0	1
0	0.6783	0.6451
1		0.4619
Cum. Avg.	0.6783	1.1070 0.8927

(a) (1,2)-Allocation.

	0	1
0	0.2734	0.2250
1	0.4333	0.3316
2		0.5827
Cum. Avg.	0.7067	1.1393 0.9230

(b) (2,3)-Allocation.

	0	1	2
0	0.6717	0.3320	0.2344
1		0.5756	0.3423
2			0.6035
Cum. Avg.	0.6717	0.9076 0.9198	1.1802

(c) (1,2,3)-Allocation.

	0	1	2	3
0	0.6623	0.3277	0.2041	0.1653
1		0.5713	0.2815	0.2149
2			0.4538	0.3069
3				0.5124
Cum. Avg.	0.6623	0.8990	0.9394 0.9250	1.1994

(d) (1,2,3,4)-Allocation.

	Time level
Symbol level	Subchannel rates
Cum. Avg.	Column-wise sums Average rate

(e) Table legend.

Table 5.7: Achievable rates for the binary subchannels, obtained using various non-uniform subchannel allocations. $E_s/N_0 = 7$ dB, blocklength: 10^5 samples.

creases, the proposed communication scheme will lose its attractiveness because of the exponential increase in computational complexity.

5.2.2 Probability densities of subchannel log likelihood ratios

As explained in Chapter 2, the low-density parity-check component codes can be optimized for the statistical behavior in their subchannel. This subchannel behavior is defined by the probability density function of the log likelihood ratios (LLRs). Like Soriaga et al. (2007), we obtain the subchannel LLR densities by sampling with a Monte Carlo simulation.

We have run such simulations with two different setups: two time levels, each having two symbol levels (two-bit binomial superposition modulation) and three time levels, each having three symbol levels (three-bit binomial superposition modulation). By using the level allocation notation defined in Section 5.2.1, we abbreviate these two setups by (2,2) and (3,3,3) respectively. We have made histograms (each having 2048 bins) of the LLR values, which serve as approximations of the probability densities. The histograms of the (2,2) setup are plotted in Figure 5.5, Figure 5.6 shows the histograms belonging to the (3,3,3) setup. In both figures, the alphabetically ordered subfigure captions follow the decoding order. The shape and location on the horizontal axis of these distributions provide insight into the multistage decoding process. In both figures, we can see that the cumulative probability for positive LLRs (corresponding to correct information about a bit) increases as number of decoded levels increases. The variation of the densities between the different symbol levels nicely show the superposition modulation concept. Also, the influence of the intersymbol interference can be seen in the figure. E.g. in Figure 5.6–g, the intrinsically bimodal density initially appears as a unimodal density (Figure 5.6–a).

5.2.3 LDPC component code design

We have optimized LDPC codes for the (2,2) setup, based on the LLR densities that were shown in the preceding subsection. The design procedure of the four component codes involved multiple iterations between code creation and performance benchmarking, to align the codes' waterfall regions as well as possible. This alignment is important because the bit-error rate (BER) versus SNR curve of the entire system will be closest to the worst performing LDPC code. The optimization results for the four component codes are shown in Table 5.8 to 5.11 in the form of normalized degree distributions from a node perspective (Richardson and Urbanke, 2007). We have benchmarked the performance of the individual codes by assuming perfect decoding of the preceding levels. The results are shown in Figure 5.7. The results have been obtained using sum-product decoding with a maximum of 800 decoding iterations.

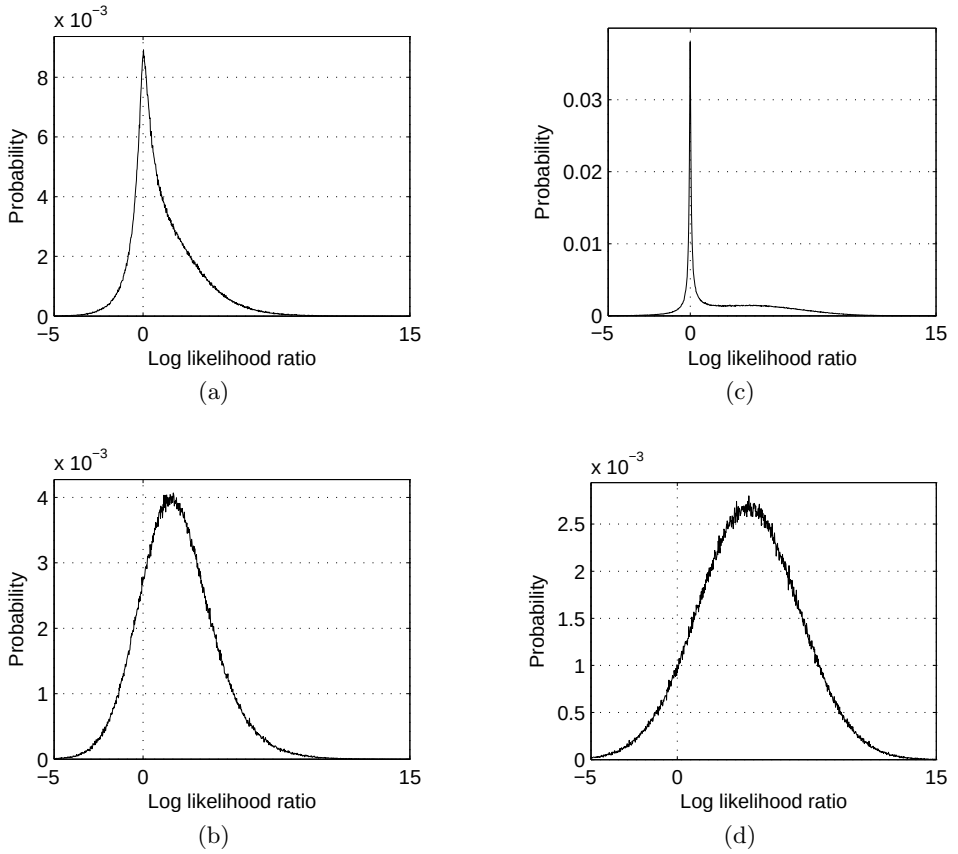


Figure 5.5: The LLR densities for each subchannel of a system with two time levels and two symbol levels per time level. The rows correspond to the symbol levels, the columns correspond to the time levels. For this simulation, we used a blocklength of 10^6.

Rate:		0.265624
Threshold found:		1.03176
Iterations required:		2000
Asympotic BER:		9.84202×10^{-9}
Threshold entropy:		0.73099
Threshold performance:		98.7413
Degree distribution pair:		
$L(x)$	$=$	$0.545269x^2 + 0.259046x^3 + 0.0556792x^6 + 0.0885713x^7 +$ $0.0392243x^{20} + 0.00217691x^{21} + 0.0100341x^{100}$
$R(x)$	$=$	$0.6608x^6 + 0.3392x^7$

Table 5.8: Information about the optimized LDPC component code for level 0.

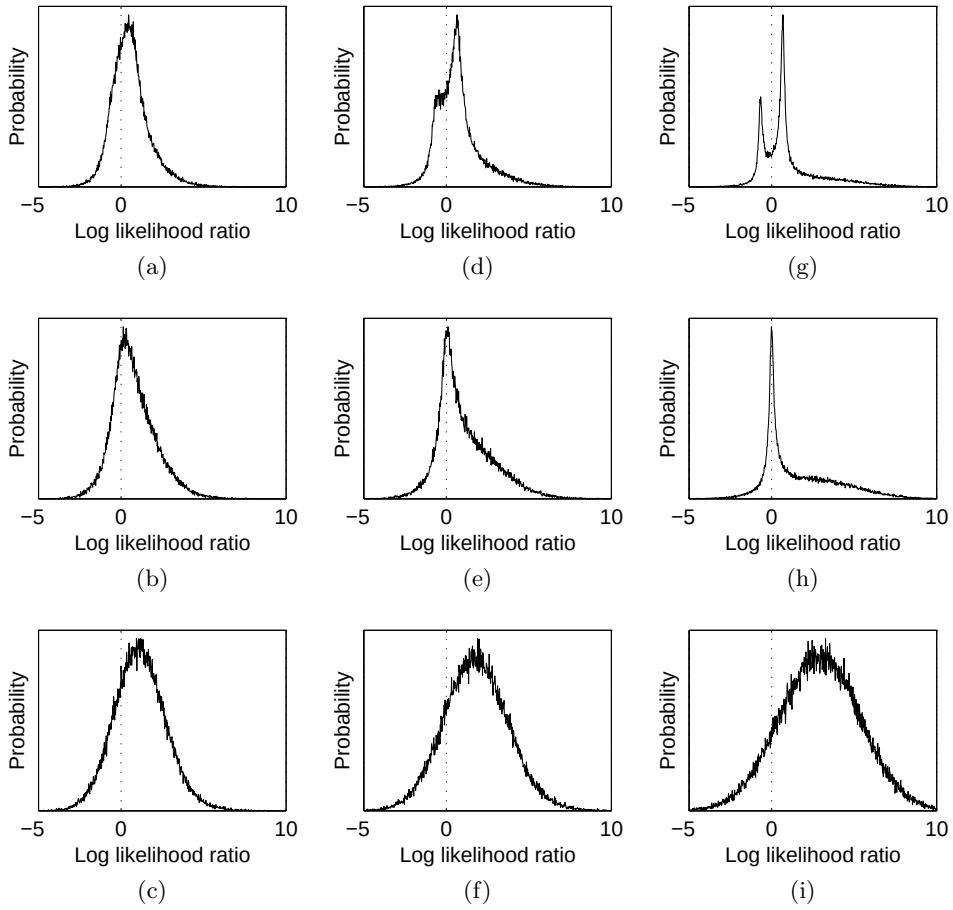


Figure 5.6: The LLR densities for each subchannel of a system with three time levels and three symbol levels per time level. The rows correspond to the symbol levels, the columns to the time levels, respectively. In order to save space, the vertical (probability) tickmarks have been omitted. For this simulation, we used a blocklength of 10^5; note the slightly bigger variance compared to Figure 5.5.

Rate:		0.451368
Threshold found:		0.998645
Iterations required:		1998
Asympotic BER:		9.76492×10^{-9}
Threshold entropy:		0.545559
Threshold performance:		99.3237
Degree distribution pair:		
$L(x)$	=	$0.47832x^2 + 0.294587x^3 + 0.0137156x^6 + 0.147912x^7 +$ $0.000515156x^{16} + 0.0172465x^{17} + 0.00812749x^{23} +$ $0.0267124x^{24} + 0.0128632x^{100}$
$R(x)$	=	$0.204981x^9 + 0.795019x^{10}$

Table 5.9: Information about the optimized LDPC component code for level 1.

Rate:		0.370909
Threshold found:		1.03011
Iterations required:		1997
Asympotic BER:		9.84552×10^{-9}
Threshold entropy:		0.625885
Threshold performance:		99.143
Degree distribution pair:		
$L(x)$	=	$0.526269x^2 + 0.269276x^3 + 0.0446731x^6 + 0.103182x^7 +$ $0.00797762x^{19} + 0.0366262x^{20} + 0.0119958x^{100}$
$R(x)$	=	$0.15637x^7 + 0.84363x^8$

Table 5.10: Information about the optimized LDPC component code for level 2.

Rate:		0.72159
Threshold found:		0.99894
Iterations required:		1975
Asympotic BER:		9.66517×10^{-9}
Threshold entropy:		0.276233
Threshold performance:		99.6993
Degree distribution pair:		
$L(x)$	=	$0.395469x^2 + 0.343096x^3 + 0.123772x^7 + 0.0619436x^8 +$ $0.000400213x^{20} + 0.0547228x^{23} + 0.00325315x^{24} +$ $0.0173429x^{100}$
$R(x)$	=	$0.510904x^{22} + 0.489096x^{23}$

Table 5.11: Information about the optimized LDPC component code for level 3.

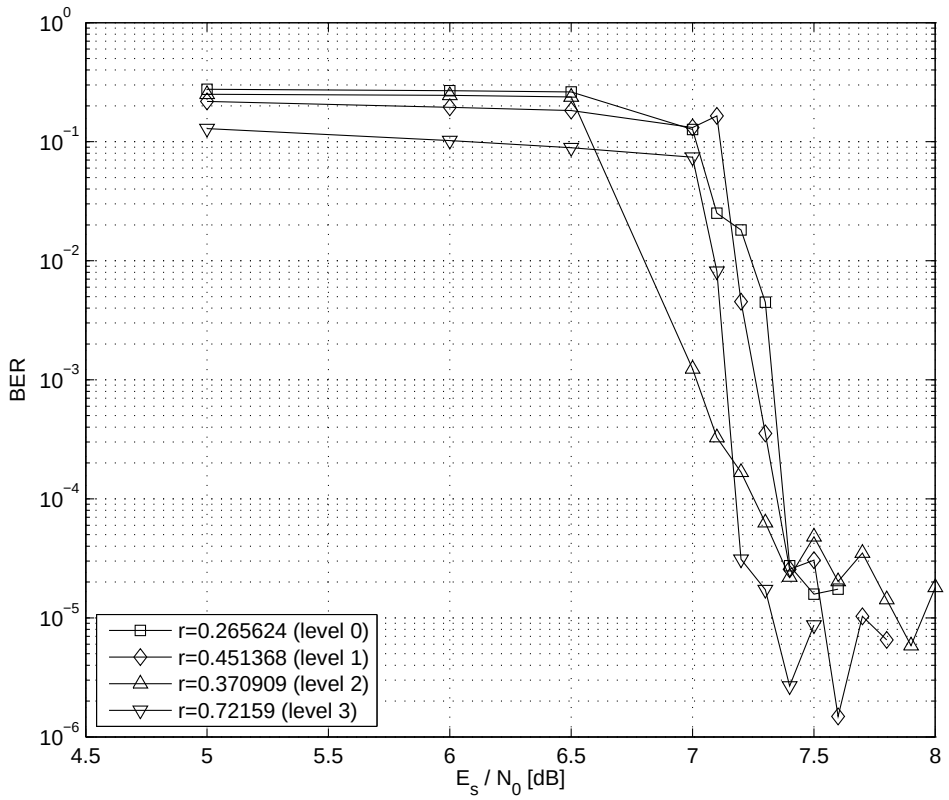


Figure 5.7: Individual performances of the optimized LDPC component codes, in the form of BER-versus-SNR plots.

5.2.4 Overall system performance

The average achieved rate of the communication system is computed as the sum of the component code rates, divided by the number of time levels. In case of our design example, we have two time levels and the rates of the LDPC component codes can be found in Table 5.8 to 5.11. Hence, the rate of the example system is

$$\begin{aligned} R &= \frac{1}{2}(0.265624 + 0.451368 + 0.370909 + 0.72159) \\ &= 0.9047 \text{ [bits/use]}. \end{aligned} \tag{5.25}$$

If we substitute this rate into the spectral efficiency formula of the *RC* channel – Equation (5.20) – and solve for the signal-to-noise ratio, we find $E_s/N_0 = 6.0$ dB. We will now determine the lowest SNR that is needed to achieve a sufficiently low bit-error rate (around 10^{-5}). Using this figure, we can express how far away from capacity (in terms of dB's) the system operates reliably.

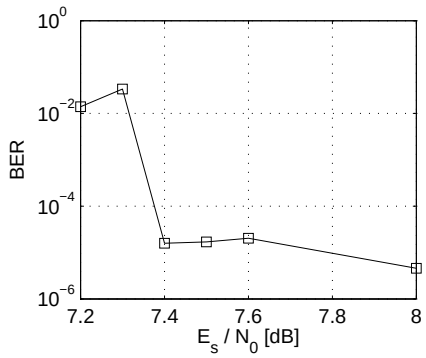
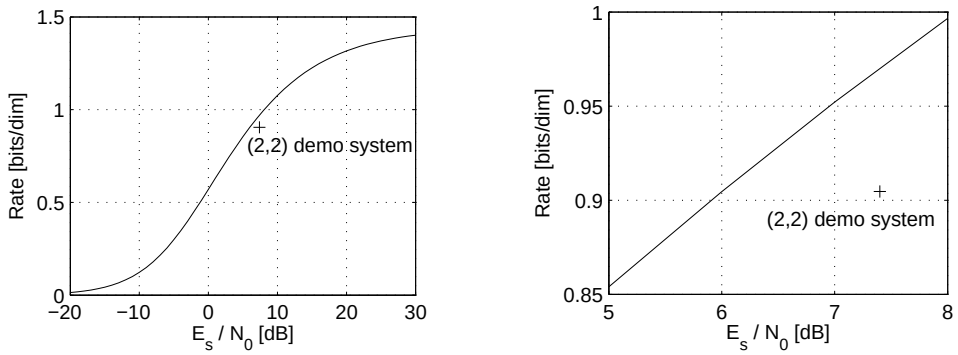


Figure 5.8: BER-versus-SNR plot for the entire system.

E_s/N_0 [dB]	A^2	Overall BER
7.2	1.98965	0.0139147
7.3	2.036	0.0335542
7.4	2.08342	1.58663e-05
7.5	2.13195	1.68812e-05
7.6	2.18161	2.0297e-05
8.0	2.39209	4.57921e-06

Table 5.12: BER-versus-SNR simulation for the entire system.

The performance of the entire system, using realistic side information from previous levels (i.e. also simulating error propagation effects) is shown in Figure 5.8, Table 5.12 and also in Figure 5.9. It can be seen from the data that the waterfall region of the system lies somewhere between $E_s/N_0 = 7.3$ dB and 7.4 dB. Hence, the system operates 'reliably' at $7.4 - 6.0 = 1.4$ dB away from capacity.



(a) Overview.

(b) Zoomed in.

Figure 5.9: Performance of the (2,2) demo system plotted together with the capacity curve of the RC channel.

Chapter 6

Concluding Remarks and Suggestions for Further Research

6.1 Conclusions

In this project, we focus on spectral-efficient communication over the linear Gaussian channel. The channel possesses memory, causing intersymbol interference. Instead of using conventional multicarrier techniques like OFDM, we take an information-theoretic approach to the communication problem. By this we mean that we apply good error-correcting codes, combined with multilevel coding and multistage decoding. We do not try to combat intersymbol interference; we introduce even more intersymbol interference by shaping the input power spectral density to the desired water-pouring PSD to be power-efficient at the transmitter. At the receiver, the ISI is accounted for in the APP detectors. We assume a scenario in which the channel is stationary in time and its characteristics are known at the transmitter. In the report, we deal with serial transmission of real-valued symbols, but it is straightforward to extend the work to transmission of complex symbols.

We present existing methods to compute the water-pouring capacity as well as achievable information rates (given an arbitrary input distribution) of the Gaussian channel. Also, we briefly describe an existing method to convert a continuous-time channel transfer function into an equivalent discrete-time description.

We extend the binary communication scheme of Soriaga *et al.* to higher order signal constellations. We apply superposition modulation and use a binomial signal constellation that achieves shaping gain on the linear Gaussian channel. The binomial constellations outperform ordinary PAM constellations.

The proposed communication scheme is demostrated on an RC lowpass network, which is an instance of the linear Gaussian channel. For this particular channel, the choice of the assumptions (a power constraint, but no

bandwidth constraint) makes that the spectral efficiency goes to a constant for infinite SNR.

For the RC channel at $E_s/N_0 = 7$ dB, we design a communication system with two time levels and two symbol levels per time level. This system operates reliably (BER $\approx 10^{-5}$) at $E_s/N_0 = 7.4$ dB, at a rate that is equal to the water-filling capacity at $E_s/N_0 = 6.0$ dB. Hence, the system operates at 1.4 dB away from capacity. At this moderate target SNR, non-uniform level allocations do not increase the efficiency of the system.

The complexity of the APP detector scales exponentially in the length of the discrete-time channel model as well as in the number of symbol levels. This means that the proposed scheme becomes computationally unattractive at high spectral efficiencies.

6.2 Further research

In this section, we present some ideas for further investigation in bulleted form.

- Adding a bandwidth constraint to the working example (the RC network) will change the bandwidth efficiency curve as a function of SNR. This will allow us to experiment with higher bandwidth efficiencies.
- Experiment with trade-offs between the number of symbol levels and the truncation length of the FIR channel model that is used in the BCJR algorithm.
- Investigate the potential of a non-uniform allocation of symbol levels over time levels at higher spectral efficiencies.
- Find expressions for the number of additions and multiplications, as a function of the level allocation.
- Compare, for a particular scenario, the proposed scheme to OFDM, the commonly used multicarrier transmission scheme, in terms of performance and 'cost' (bandwidth, power, computational complexity).

References

S. Aji and R. McEliece. The generalized distributive law. *IEEE Trans. on Inf. Theory*, 46(2), 2000.

D. Arnold and H.-A. Loeliger. On the information rate of binary-input channels with memory. In *Proc. of IEEE International Conference on Communications (ICC)*, pages 2692–2695, Helsinki, Finland, June 2001.

D. Arnold, H.-A. Loeliger, and P. O. Vontobel. Computation of information rates from finite-state source/channel models. In *Proc. 40th Annual Allerton Conference on Communication, Control and Computing*, 2002.

L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Trans. on Inf. Theory*, pages 284–287, March 1974.

S. Y. Chung. *On the Construction of Some Capacity-Achieving Coding Schemes*. PhD thesis, Massachusetts Institute of Technology, 2000.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, New Jersey, 2006.

H. S. Cronie. Superposition coding for power- and bandwidth efficient communication over the gaussian channel. In *Proc. IEEE International Symposium on Information Theory*, 2007.

G. D. Forney and G. Ungerboeck. Modulation and coding for linear gaussian channels. *IEEE Trans. on Inf. Theory*, 44(6), Oct. 1998.

R. G. Gallager. *Information Theory and Reliable Communication*. Wiley, New York, 1968.

S. H. Gerez. *Hints on Report Writing*. University of Twente, Enschede, The Netherlands, 1998.

W. Hirt and J. L. Massey. Capacity of the discrete-time gaussian channel with intersymbol interference. *IEEE Trans. on Inf. Theory*, 34:380–388, 1988.

H. Imai and S. Hirakawa. A new multilevel coding method using error correcting codes. *IEEE Trans. on Inf. Theory*, 23:371–377, May 1977.

A. Papoulis. *Signal Analysis*. McGraw-Hill, 1977.

H. D. Pfister, J. B. Soriaga, and P. H. Siegel. On the achievable information rates of finite state isi channels. In *Proceedings IEEE Global Communications Conference*, 2001.

J. G. Proakis. *Digital Communications*. McGraw-Hill, New York, 2001.

T. J. Richardson and R. L. Urbanke. Modern coding theory. URL `http://lthcwww.epfl.ch/papers/mct.pdf`. [Online manuscript], 2007.

R. Rojas. The kalman filter. Technical report, Freie Universität Berlin, Institut für Informatik, 2003.

C. E. Shannon. A mathematical theory of communication. *Bell Sys. Tech. J.*, 27:379–423, 623–656, 1948.

V. Sharma and S. Singh. Entropy and channel capacity in the regenerative setup with applications to markov channels. In *Proc. IEEE Int. Symp. on Inform. Theory*, page 283, Washington, DC, USA, June 2001.

S. S. Soliman and M. D. Srinath. *Continuous and Discrete Signals and Systems*. Prentice Hall, New Jersey, 1998.

J. B. Soriaga, H. D. Pfister, and P. H. Siegel. Determining and approaching achievable rates of binary intersymbol interference channels using multistage decoding. *IEEE Trans. on Inf. Theory*, 53(4):1416–1429, April 2007.

S. Yang and A. Kavčić. Capacity of partial response channels. In B. Vasic and E. M. Kurtas, editors, *Coding and Signal Processing for Magnetic Recording Systems*. CRC Press, Boca Raton, 2005.