Bachelor Project

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Radio Engineering

# Wireless Physical Layer Network Coding in Simple Network Topologies with Realistic Channel Models

Richard Elsner

Supervisor: prof. Ing. Jan Sýkora, CSc.
May 2022

ČVUT
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

## I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Elsner** Jméno: **Richard** Osobní číslo: **491839**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra radioelektroniky**

Studijní program: **Otevřené elektronické systémy**

## II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

**Bezdrátové síťové kódování fyzické vrstvy v jednoduchých síťových topologiích s realistickými modely kanálu**

Název bakalářské práce anglicky:

**Wireless Physical Layer Network Coding in Simple Network Topologies with Realistic Channel Models**

Pokyny pro vypracování:

Student will first get acquainted with with fundamentals of modulation, coding and demodulation/decoding methods of Wireless Physical Layer Network Coding (WPNC), including basic methods for carrier and timing synchronisation. The goal of the work is to design, analyse and functionally verify the selected WPNC communication scenario. This should include suitably selected WPNC strategy (e.g. hierarchical decode and forward strategy using layered NCM with hierarchical demodulation/decoding) in simple topologies (e.g. two-way relay channel, or butterfly network) and with a strong focus on realistic channel parametrisation model. The functional verification should be done by means of computer simulation and, if possible for some selected parts of the system, by a practical implementation test using over-the-air test-bed.

Seznam doporučené literatury:

[1] J. Sykora, A. Burr: Wireless Physical Layer Network Coding, Cambridge University Press 2018
[2] J. G. Proakis: Digital communications, 4th ed. 2001

Jméno a pracoviště vedoucí(ho) bakalářské práce:

**prof. Ing. Jan Sýkora, CSc. katedra radioelektroniky FEL**

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **27.01.2022** Termín odevzdání bakalářské práce: **20.05.2022**

Platnost zadání bakalářské práce: **30.09.2023**

| | | |
|---|---|---|
| prof. Ing. Jan Sýkora, CSc.<br>podpis vedoucí(ho) práce | doc. Ing. Stanislav Vítek, Ph.D.<br>podpis vedoucí(ho) ústavu/katedry | prof. Mgr. Petr Páta, Ph.D.<br>podpis děkana(ky) |

## III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

Datum převzetí zadání Podpis studenta

# Acknowledgements

I wish to express my thanks to my supervisor, prof. Ing. Jan Sýkora, CSc. for his guidance and for providing the materials for the WPNC. I would also like to thank all my close ones for their support throughout my studies.

# Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

In Prague, date . . . . . . . . . . . . . . . .

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne . . . . . . . . . . . . . . . . .

# Abstract

This bachelor's thesis goal is the application of the WPNC and estimation and detection theory principles in a simple network topology.

The reader will first get acquainted with the necessary theoretical fundamentals, then with the case for the topology "Two-Way Relay Channel". Next chapter demonstrates the practical implementation in the program Matlab. The last chapter focuses on the numerical results of the said program.

**Keywords:** WPNC, BPSK, QPSK, Two-Way Relay Channel

**Supervisor:** prof. Ing. Jan Sýkora, CSc.

# Abstrakt

Cílem této bakalářské práce je aplikování principů WPNC a teorie detekce a estimace v jednoduché komunikační topologii.

Čtenář bude nejdříve seznámen s nezbytnými teoretickými základy, následně s problematikou pro příklad u topologie "Two-Way Relay Channel". Poté dojde k demonstraci praktické implementace v programu Matlab. Poslední kapitola se zabývá výsledky výpočtů v již zmíněném programu.

**Klíčová slova:** WPNC, BPSK, QPSK, Two-Way Relay Channel

**Překlad názvu:** Bezdrátové síťové kódování fyzické vrstvy v jednoduchých síťových topologiích s realistickými modely kanálu

# Contents

**1 Introduction** **1**

1.1 Introduction to digital communication . . . 2

1.1.1 Digital communication scheme 2

1.1.2 Signal properties . . . 3

1.2 Encoding . . . 4

1.2.1 Convolutional code . . . 4

1.3 Modulation . . . 4

1.3.1 General principles . . . 5

1.3.2 Modulation pulses . . . 5

1.3.3 PSK modulation . . . 6

1.4 Communication channel . . . 7

1.5 Demodulation . . . 8

1.5.1 MAP and ML receiver . . . 8

1.5.2 Decision regions . . . 9

1.5.3 Minimum distance detector . . 10

1.5.4 Matched filter . . . 11

1.6 Decoding . . . 12

1.6.1 Viterbi algorithm . . . 12

1.7 Estimation theory . . . 13

1.7.1 Cramer-Rao Lower Bound . . 14

1.7.2 Maximum-Likelihood Estimator . . . 15

1.8 Detection Theory . . . 16

1.8.1 Neyman-Pearson Theorem . . 16

1.9 Introduction to WPNC . . . 17

1.9.1 2-Way Relay Channel . . . 17

1.9.2 Hierarchical decode and forward . . . 17

1.9.3 H-constellation . . . 18

**2 WPNC in 2WRC with realistic channel model** **21**

2.1 General overview . . . 22

2.2 Coding . . . 22

2.3 Signal modulation . . . 23

2.4 Communication channel . . . 23

2.5 Relay node processing . . . 24

2.5.1 Neyman-Pearson Theorem . . 24

2.5.2 Estimating the channel parameters . . . 25

2.5.3 Data detecting . . . 27

2.5.4 Broadcasting the detected data 28

2.6 Processing at the sources . . . 29

2.7 Decoding . . . 30

2.7.1 Single signal case . . . 31

2.7.2 Multiple signals case . . . 31

**3 Implementation** **33**

3.1 Sources transmitting . . . 34

3.1.1 Encoding . . . 34

3.1.2 Modulation . . . 34

3.1.3 Channel model . . . 35

3.2 Processing at the relay node . . . 35

3.2.1 Pilot signal detection . . . 35

3.2.2 Parameters estimation . . . 35

3.2.3 Case for single source transmitting . . . 36

3.2.4 Case for both signals transmitting . . . 36

3.3 Relay node broadcasting . . . 38

3.3.1 Case for single source transmitting . . . 38

3.3.2 Case for both sources transmitting . . . 38

3.4 Processing at the sources . . . 38

3.4.1 Hard decision data detection 40

3.4.2 Decoding . . . 40

**4 Simulation and results** **43**

4.1 Signal detection . . . 44

4.2 Channel parameter estimation . . 45

4.2.1 Delay estimation . . . 45

4.2.2 Phase estimation . . . 46

4.3 Error performance . . . 49

4.3.1 SNR dependence . . . 49

4.3.2 Relative fading dependence . . 51

4.4 Imperfect symbol period synchronization . . . 53

**5 Conclusion** **55**

**Bibliography** **57**

**A Attached Matlab files** **59**

# Figures

1.1 Digital communication scheme . . . 2
1.2 Implementation of the convolutional coding . . . 4
1.3 RRC pulse in time domain . . . 6
1.4 PSK modulations . . . 6
1.5 Channel model . . . 7
1.6 LTI AWGN channel . . . 7
1.7 Decision regions . . . 9
1.8 Decision regions of PSK modulations . . . 11
1.9 Implementation of the matched filter . . . 11
1.10 Example of Viterbi algorithm . . . 12
1.11 Different PDFs for different hypohesis . . . 13
1.12 ML estimator scheme . . . 15
1.13 Two-Way Relay Channel topology . . . 17
1.14 Comparison between TDMA and WPNC . . . 18
1.15 H-constellation for two BPSK modulations . . . 18
1.16 H-constellation for BPSK modulation for different values of $h$ 20

2.1 Implemented code . . . 23
2.2 Scheme of the data modulation . . . 23
2.3 General scheme for relay node processing . . . 24
2.4 Scheme of Joint ML estimator . . . 26
2.5 Scheme of data detection on relay node . . . 28
2.6 Data broadcasted by relay node if multiple sources transmitted . . . 30
2.7 Processing at the sources . . . 31

3.1 Implementation of the function *code* . . . 34
3.2 Implementation of the function *Neyman_Pearson* . . . 36
3.3 Implementation of the function *parameters* . . . 37
3.4 Finding closest H-Constellation point in the function *estimate_data_from_both_sources* for BPSK modulation . . . 39
3.5 Implementation of the function *Viterbi* for BPSK modulation . . . 42

4.1 Correct detection dependence on SNR and $\alpha$ . . . 44
4.2 Accuracy of delay estimation using Joint ML estimator . . . 45
4.3 Phase estimation histograms compared with ideal distribution function for single CAZAC pilot . . . 47
4.4 Phase estimation histograms for SNR = 20 dB . . . 48
4.5 Phase estimation histograms for SNR = 3 dB . . . 48
4.6 Error performance dependence on SNR . . . 50
4.7 Error performance dependence on angle of relative fading $h$ for QPSK modulation . . . 51
4.8 H-Constellations for certain values of $h$ for QPSK modulation . . . 52
4.9 Effect of the imperfect synchronization on error performance . . . 54

# Tables

2.1 XOR function in constellation space for BPSK modulation . . . . . . 32
2.2 XOR function in constellation space for QPSK modulation . . . . . . 32

4.1 Relation of the SNR to the standard deviation for single pilot case . . . . . . . . . . . . . . . . . . . . . . . . . . 46
4.2 Relation of the SNR to the standard deviation for multiple pilot case . . . . . . . . . . . . . . . . . . . . . . . . . . 48

# Chapter 1

# Introduction

This thesis deals with the WPNC and detection and estimation theory fundamentals applied to simple network topology, "Two-Way Relay Channel."

The communication between two sources is conveyed with the help of a relay node. The relay node can broadcast the coded information containing data from both sources even when both sources are not perfectly synchronized. The synchronization imperfection amounts to integers of the symbol period. As the sources are unaware of what the imperfection amounted to, the relay node will transmit a header as part of the data payload containing all the additional information the sources need to decode the data correctly.

Each data payload starts with modulated CAZAC sequence that helps with detection and estimation. The nodes will firstly use the Neyman-Pearson theorem to detect the modulated CAZAC sequence in the received signal. After the signal is assumed to be present, the nodes will use the Maximum-Likelihood estimators to estimate the following channel parameters: a random phase shift and delay equal to the integers of the symbol period. With the knowledge of the parameters, the signal processing can happen.

In the first chapter, the needed theoretical background is established. Starting with the digital communication basics, we will touch on some fundamentals regarding the estimation and detection theory and end with a light introduction to the basics of WPNC.

## 1.1 Introduction to digital communication

### 1.1.1 Digital communication scheme

This section will briefly describe every important function block in the digital communication scheme and its function in transmitting information in the form of a binary sequence from source to destination. In later sections, each block will be covered more in-depth, along with practical implementation. This introduction is inspired by [1, p. 1 - 3].

Firstly the information itself in the form of a binary sequence noted as $\boldsymbol{d}$ goes to the *encoder*, where it is converted into a "better," more resistant sequence that makes it easier for the receiver to overcome the effects of channel parameters when decoding the data. The encoder does so by adding redundancy to binary information sequence and thus creates a longer sequence $\boldsymbol{c}$.

Secondly the encoded sequence $\boldsymbol{c}$ passes to the *digital modulator* that maps the sequence into continuous-time signal wave-forms $s(t)$ using chosen modulation pulse with desired characteristics.

Then the signal transmits through the communication ether via the *communication channel*. The channel is corrupted by random variables and lowers the quality of the transmitted signal. Its effects are described using input-output models.

The corrupted signal $x(t)$ then gets to the receiver's end, where it gets processed in a *digital demodulator* and *decoder* from which we get the message estimation $\hat{\boldsymbol{d}}$. The hat notation will denote estimation as used in [5] and [6].

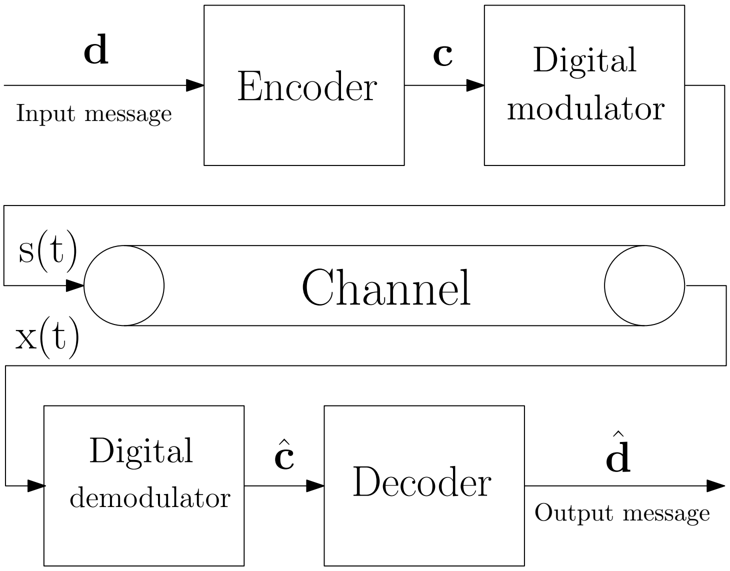


**Figure 1.1:** Digital communication scheme

### 1.1.2 Signal properties

Here some essential terms around signals and their properties will be defined. The terms are *signal energy* and a signal *vector space representation*. The definitions are taken from [1, p. 25, 26] and [1, p. 30, 31] respectively.

#### Signal energy

The definition of continuous-time signal $s(t)$ energy is in equation 1.1.

$$\epsilon_s = \int_{-\infty}^{\infty} |s(t)|^2 \mathrm{d}t \tag{1.1}$$

We can also define the cross-energy of two signals as their *inner product* 1.2.

$$\langle s_1(t), s_2(t) \rangle = \int_{-\infty}^{\infty} s_1(t) s_2^*(t) \mathrm{d}t = \epsilon_{s_1,s_2} \tag{1.2}$$

#### Vector space representation

In this part I will show that we can represent continuous-time signal $s(t)$ as a vector $\boldsymbol{s}$ with all its properties remained intact. This representation makes the signal processing much easier.

Let us have a set of signals called *basis signals* $\{\xi_i(t)\}_{i=0}^{N}$. We say that this set is **orthogonal** if the cross-energy of any two different signals is zero 1.3.

$$\langle \xi_i(t), \xi_{i'}(t) \rangle = \int_{-\infty}^{\infty} \xi_i(t) \xi_{i'}^*(t) \mathrm{d}t = 0 \quad \forall i \neq i' \tag{1.3}$$

We can then represent our own signal as a linear combination of these signals 1.4.

$$s(t) = \sum_{i=0}^{N} s_i \xi_i(t) \tag{1.4}$$

Each constant $s_i$ is equal to the inner product of the signal $s(t)$ and the basis signal $\xi_i(t)$ 1.5.

$$s_i = \langle s(t), \xi_i(t) \rangle \tag{1.5}$$

## 1.2 Encoding

The goal of an encoder is to somehow transform the binary information stream into another sequence that makes it easier for the receiver to decrypt the original information even when the transmission of the information is corrupted, typically by noise. One such type of coding that will be touched upon is *convolutional coding*. The following definitions and relations are taken from [1, p. 491, 492].

### 1.2.1 Convolutional code

The convolutional code is obtained by passing the message sequence through a linear finite shift register. At any time slot, the machine operates with $k$-bits long sub-sequence of the whole sequence $\boldsymbol{d}$ and produces an $n$-bits long encoded sequence. The *code rate* of such register is defined as $R_c = \frac{n}{k}$.

The convolutional encoder can be fully described by its *function generators* $\boldsymbol{g}_i$. The output of such function generator is an inner product with the data sequence of length $k$. In general the convolutional encoder has $n$ function generators $\boldsymbol{g}_i$ that can be written in a $n \times k$ matrix $\boldsymbol{G}$.

An example of the generator matrix $\boldsymbol{G}$ is in equation 1.6 and the relation between coded data $\boldsymbol{c}$ and data bits $\boldsymbol{d}$ is in equation 1.7. Lastly the implementation scheme of the coding is in figure 1.2.

$$\boldsymbol{G} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \tag{1.6}$$

$$\begin{aligned} c_{n,1} &= d_{n-1} + d_n \\ c_{n,2} &= d_n \end{aligned} \tag{1.7}$$

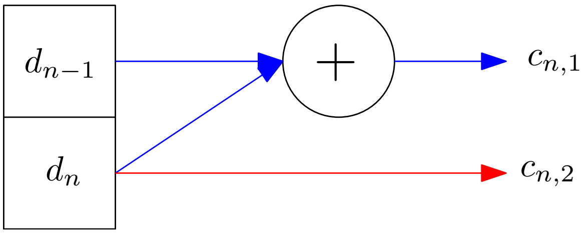


**Figure 1.2:** Implementation of the convolutional coding

## 1.3 Modulation

Now let us take a look at general principles of modulation in digital communication systems, the modulation pulses, their desired properties, and lastly, at one such type of modulation in more detail. The relations are taken from [5, presentations 01 and 02].

### 1.3.1 General principles

Digital modulation in its core is a transformation of discrete binary and in our case encoded data symbols $\mathbf{c} = [..., c_n, ...]$ into a continuous-time signal $s(t)$. The relation between between symbols and the signal is described in 1.8.

$$s(t) = \sum_n g(c_n, \sigma_n, t - nT_s) = \sum_n g(q_n, t - nT_s) \tag{1.8}$$

The $T_s$ is a symbol period, $c_n$ is $n$-th encoded data symbol, $\sigma_n$ is $n$-th modulator state, $q_n$ is $n$-th channel symbol that depends on data symbol and modulator state $q_n = q(c_n, \sigma_n)$. Lastly $g(t)$ is a continuous-time modulation pulse.

For memoryless modulation, the modulator state becomes useless because there is no memory between data symbols, and therefore the relation for channel symbol $q_n$ simplifies into $q_n = q(c_n)$.

Another simplification is using a linear modulation. In linear modulation, the modulation pulse $g(t)$ depends only on time and is multiplied by channel symbol $q_n$. With this, we can rewrite the relation into the equation 1.9.

$$s(t) = \sum_n q_n(c_n) g(t - nT_s) \tag{1.9}$$

### 1.3.2 Modulation pulses

As we could see in the previous equation, the overall continuous-time signal a linear memoryless modulator creates is a simple sum of scaled time-shifted modulation pulses. However, there is one condition the pulse must satisfy so that the differently time-shifted signals do not affect each other. We will call this condition a *Nyquist condition.*

The modulation pulse $g(t)$ is Nyquist if the relation 1.10 is satisfied. Simply said, the pulse $g(t)$ is Nyquist if any two differently time-shifted pulses are orthogonal.

$$\int_{-\infty}^{\infty} g(t - nT_s) g^*(t - mT_s) = 0, \quad \forall n \neq m \tag{1.10}$$

An example of one such modulation pulse satisfying the said condition is a Root Raised Cosine Pulse defined in 1.11. The pulse is plotted in the time-domain in figure 1.3.

$$g_{RRC}(t) = \begin{cases} \frac{1}{\sqrt{T_s}\left(1-16\alpha^2 \frac{t^2}{T_s^2}\right)} \left( \frac{\sin\left((1-\alpha)\frac{\pi t}{T_s}\right)}{\frac{\pi t}{T_s}} + \frac{4\alpha\left((1+\alpha)\frac{\pi t}{T_s}\right)}{\pi} \right) & \\ \frac{1}{\sqrt{T_s}} \left(1 - \alpha + \frac{4\alpha}{\pi}\right) & \text{for } t = 0 \\ \frac{\alpha}{\pi\sqrt{2T_s}} \left((\pi - 2)\cos\left(\frac{\pi}{4\alpha}\right) + (\pi + 2)\sin\left(\frac{\pi}{4\alpha}\right)\right) & \text{for } t = \pm\frac{T_s}{4\alpha} \end{cases} \tag{1.11}$$

RRC pulse is also scaled to have unit energy 1.12. This scaling helps in computation where we do not have to re-scale the results.

$$\int_{-\infty}^{\infty} |g_{RRC}(t)|^2 \mathrm{d}t = 1 \tag{1.12}$$

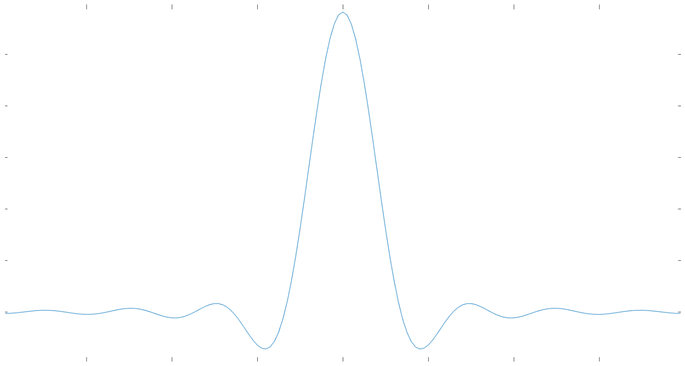


**Figure 1.3:** RRC pulse in time domain

### 1.3.3 PSK modulation

So far, we have talked about a general modulation. Now a very commonly used linear modulation will be mentioned - a Phase Shift Keying modulation (PSK for short).

A PSK modulation maps the data symbols into constellation points located on a unit circle in a complex plane. For general $M_q$-PSK ($M_q \in \mathbb{N}$) modulation, we can define the channel symbols in constellation space as 1.13.

$$q_i \in \{e^{\mathrm{j}\frac{2\pi i}{M_q}}\}_{i=0}^{M_q-1} \tag{1.13}$$

The most commonly used PSK modulations are BPSK and QPSK. The channel symbols of these two modulations are in figure 1.4 together with the binary data symbols that are mapped onto the respective channel symbols.

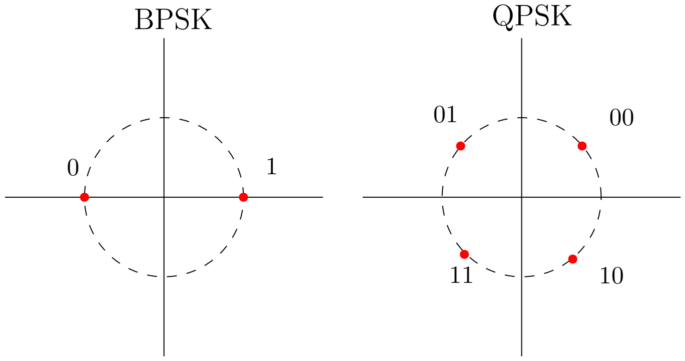


**Figure 1.4:** PSK modulations

## 1.4 Communication channel

By channel, we describe any form of change and attenuation to the useful signal $s(t)$ between the source and its destination. We can completely characterize the channel input-output relation by an equation and channel parameters that can be random or deterministic 1.5.

The source for this section is [5, presentation 08].

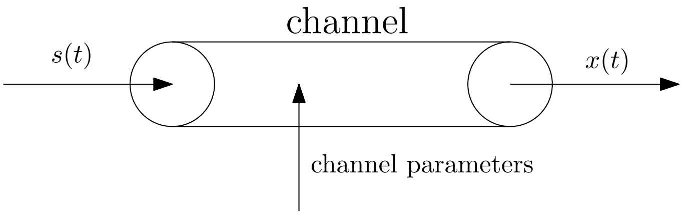


**Figure 1.5:** Channel model

Next, a widespread composite channel model called *Linear Time-Invariant channel with complex AWGN* will be introduced. AWGN stands for *"Additive White Gaussian Noise"*. The noise has the following properties:

- **White** means its spectrum in the frequency domain is a constant. As a result, the noise affects all frequencies indiscriminately.
- **Gaussian** means the noise has a Gaussian probability distribution function (PDF). If the noise is complex, we say that the real and imaginary noise values are independent and identically distributed (IID).

The input-output relation of this channel model can be described as shown in equation 1.14. The $s(t)$ is the modulated useful signal, and $x(t)$ is the noisy signal the channel has returned. The $\varphi$ represents the phase shift, and the $\tau$ is the time delay. Lastly, the $w(t)$ represents the complex AWGN in the time domain. The scheme is plotted in figure 1.6.

$$x(t) = e^{\mathrm{j}\varphi} s(t-\tau) + w(t) \tag{1.14}$$

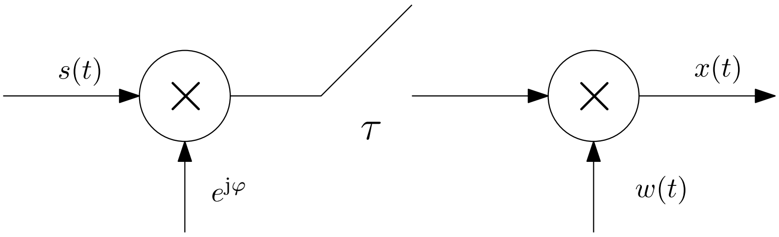


**Figure 1.6:** LTI AWGN channel

## 1.5 Demodulation

In previous sections, I have described the digital modulation used for transmitting digital information through a communication channel and the properties of one such channel. This chapter will describe the characteristics and methods of designing an optimal receiver for the corrupted signal. Here we will purely focus on receiving a signal that has been corrupted with AWGN.

The source for this section is [1, p. 160 - 179]

### 1.5.1 MAP and ML receiver

Let us have a received signal $\boldsymbol{x}$ in vector space representation that has passed through an AWGN channel with additive Gaussian noise $\boldsymbol{w}$. Next let us have a function of a receiver $g(\boldsymbol{x})$: $\mathbb{R}^N \to m \in \mathcal{A}_m$. This function takes the received signal vector and transforms it into a sent message $m$ from an alphabet $\mathcal{A}_m$. Now let us assume that the receiver decided that message $\hat{m}$ has been transmitted based on a result from the function $g(\cdot)$. The probability that this is the correct decision is the probability of $\hat{m}$ being sent. By writing this in an equation, we get 1.15.

$$P(\text{correct decision}|\boldsymbol{x}) = P(\hat{m}\text{ sent}|\boldsymbol{x}) \tag{1.15}$$

In order to get a non-conditional probability we use the probability of $\boldsymbol{x}$ being received $p(\boldsymbol{x})$. For the overall probability of receiver making the correct decision we get 1.16.

$$\begin{aligned} P(\text{correct decision}) &= \int P(\text{correct decision}|\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\ &= \int P(\hat{m}\text{ sent}|\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \end{aligned} \tag{1.16}$$

Our goal is to maximize the probability of a correct decision. The maximization is achieved by maximizing the conditional probability function in the integral above because PDF $p(\boldsymbol{x})$ is a positive function. With that we get relation 1.17 for the estimated message $\hat{m}$.

$$\hat{m} = \arg\max_{m\in\mathcal{A}_m} P(m|\boldsymbol{x}) \tag{1.17}$$

Transmitting message $m \in \mathcal{A}_m$ is equivalent to transmitting a continuous-time signal $\boldsymbol{s}_m$. With that we can rewrite the previous equation as 1.18.

$$\hat{m} = \arg\max_{m\in\mathcal{A}_m} P(\boldsymbol{s}_m|\boldsymbol{x}) \tag{1.18}$$

The decision rule given by finding the maximum probability is known as *maximum a posterior* rule (or MAP rule for short).

Using Bayes theorem we can write the equation 1.18 as 1.19

$$\hat{m} = \arg\max_{m \in \mathcal{A}_m} \frac{P(m)p(\boldsymbol{x}|\boldsymbol{s}_m)}{p(\boldsymbol{x})} = \arg\max_{m \in \mathcal{A}_m} P(m)p(\boldsymbol{x}|\boldsymbol{s}_m) \tag{1.19}$$

We can ignore the probability $p(\boldsymbol{x})$ as it has no effect on maximizing the function. To find the maximum, we need to know the prior probability of sent messages $P(m)$. Assuming this probability is equiprobable, we get an even simpler equation 1.20

$$\hat{m} = \arg\max_{m \in \mathcal{A}_m} p(\boldsymbol{x}|\boldsymbol{s}_m) \tag{1.20}$$

The function $p(\boldsymbol{x}|\boldsymbol{s}_m)$ is known as the *likelihood* of message $m$ and the receiver is called the *maximum-likelihood receiver* (or ML receiver for short).

### 1.5.2 Decision regions

In the previous section, a particular function of received signal $g(\boldsymbol{x})$ that transforms the received signal into a space of messages has been introduced. In this section, this function is described more in-depth, and a typical example is provided.

Assume received signal $\boldsymbol{x} \in \mathbb{R}^N$ and the function $g(\cdot)$ returns the message that has been the most likely sent $\hat{m} = g(\boldsymbol{x})$. We can imagine the output space (constellation space for example) being divided into certain regions $D_i, i \in \{1, 2, ..., M\}$ as shown in 1.7. Each message sent $m \in \{1, 2, ..., M\}$ has its own decision region in output space and if the received signal represented in that output space happens to fall into that specific region, the probability of receiving message $m_i$ will be bigger that probabilities of receiving any other message $m_{i'}, i' \neq i$. Mathematically we can describe the regions as 1.21.

$$D_i = \{\boldsymbol{x} \in \mathbb{R}^N : P(m_i|\boldsymbol{x}) > P(m_{i'}|\boldsymbol{x}), \quad \forall i' \in \{1, 2, ..., M\} \cap i \neq i'\} \tag{1.21}$$

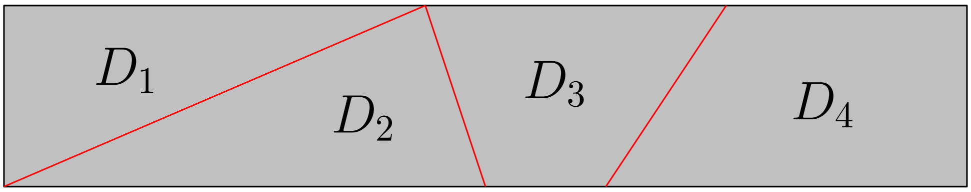


**Figure 1.7:** Decision regions

### 1.5.3 Minimum distance detector

In the last part, we have defined decision regions based on a probability. That definition gives us a basic idea though it is very vague. In this part, I will go more in-depth and give an equivalent division between each decision region based on a received signal in an AWGN channel with power spectral density $\frac{N_0}{2}$. The PDF of AWGN $\boldsymbol{w} \in \mathbb{C}^N$ is defined as 1.22.

$$p_{\boldsymbol{w}}(\boldsymbol{w}) = \left(\frac{1}{\sqrt{\pi N_0}}\right)^N e^{-\frac{\|\boldsymbol{w}\|^2}{N_0}} = \left(\frac{1}{\sqrt{\pi N_0}}\right)^N e^{-\frac{\sum_{i=1}^N |w_i|^2}{N_0}} \tag{1.22}$$

The MAP detector for the AWGN channel is described in following equations 1.23.

$$\begin{aligned}
\hat{m} &= \arg\max_{m \in \mathcal{A}_m} P(m)p(\boldsymbol{x}|\boldsymbol{s}_m) \\
&= \arg\max_{m \in \mathcal{A}_m} P(m)p_{\boldsymbol{w}}(\boldsymbol{x} - \boldsymbol{s}_m) \\
&\overset{(1)}{=} \arg\max_{m \in \mathcal{A}_m} P(m)e^{-\frac{\|\boldsymbol{x}-\boldsymbol{s}_m\|^2}{N_0}} \\
&\overset{(2)}{=} \arg\max_{m \in \mathcal{A}_m} \ln P(m) - \frac{\| \boldsymbol{x} - \boldsymbol{s}_m \|^2}{N_0} \\
&\overset{(3)}{=} \arg\max_{m \in \mathcal{A}_m} - \| \boldsymbol{x} - \boldsymbol{s}_m \|^2 \\
&\overset{(4)}{=} \arg\min_{m \in \mathcal{A}_m} \| \boldsymbol{x} - \boldsymbol{s}_m \|
\end{aligned} \tag{1.23}$$

I have used following steps in simplifying the expression for MAP detector:

1. Removing the constant $\left(\frac{1}{\sqrt{\pi N_0}}\right)^N$ because it does not depend on message $m$
2. Applying ln function on the argument. The logarithm is an increasing function, so it does not change the function's maximum.
3. Equiprobable distribution of messages $P(m)$ is assumed
4. The minus changes finding the maximum into finding the minimum of a positive function, and the square also does not affect the peak of the function

We can see that finding the message with the biggest probability of being sent is equivalent to finding the closest channel symbol representing sent message to our received signal in the constellation space.

Finding the closest channel symbol has a very straightforward geometric interpretation, as we can see in the figure 1.8. For BPSK modulation, the constellation points are $\{\pm 1\}$, so the two decision regions are divided by an imaginary axis. For the QPSK modulation, the regions are divided by both real and imaginary axis.

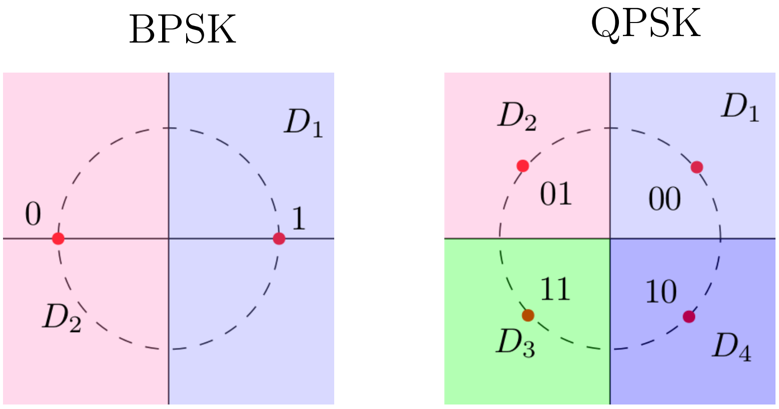


**Figure 1.8:** Decision regions of PSK modulations

We can also rewrite the equation defining decision regions 1.21 into 1.24. The $D_i$ is a decision region for message $m_i \in \mathcal{A}_m$ assuming equiprobable probability $P(m)$ of messages sent.

$$D_i = \{\boldsymbol{x} \in \mathbb{R}^N : \| \boldsymbol{x} - \boldsymbol{s}_{m_i} \| < \| \boldsymbol{x} - \boldsymbol{s}_{m_i'} \|, \forall i' \neq i\} \tag{1.24}$$

### 1.5.4 Matched filter

After the theoretical background has been established in previous sections, we can look at the practical implementation of one specific demodulator.

The demodulator that will be used throughout this work is called *matched filter*. This filter's impulse response matches the used modulation pulse - an RRC pulse $g(t)$ in our case. If we put the received noisy signal $x(t)$ as an input of the filter, we can get the constellation space representation $\boldsymbol{x}$ of the received signal by sampling the filter output at specific times. The specific times will be integers of the symbol period $T_s$ [5, presentation 09].

The implementation scheme is shown below in figure 1.9.

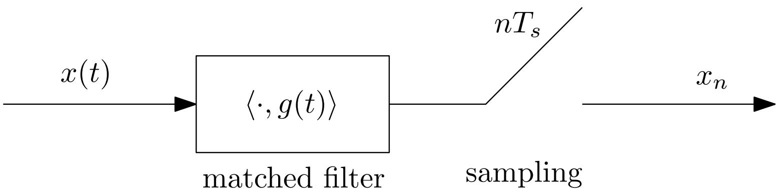


**Figure 1.9:** Implementation of the matched filter

## 1.6 Decoding

The decoder is a part of a communication scheme that transforms the data from the demodulator in the form of constellation space points into an estimation of sent data. The decoding is no easy task because the ideal decoder would need to compare the received data to all the possible data combinations the source might have sent. Then, based on its own metric $\rho$, the decoder would pick the sequence that fits the most.

This is impossible to compute with longer data sequences as the number of computation steps increases exponentially with the data length. A solution to this problem is called the *Viterbi algorithm*, a sub-optimal algorithm for finding the best approximation where the number of computation steps increases linearly with increasing length of message.

The source for this section is [5, presentation 09].

### 1.6.1 Viterbi algorithm

This algorithm solves the previously mentioned problem by assuming that the "best" way to get from source to destination is equivalent to picking the best way from $i$-th state to $(i+1)$-th state in each step. By "best" way, we mean the way that maximizes the likelihood function in that step or, in other words, minimizes the decoding metric $\rho$.

$$\rho_{\text{min}}(A,B) = \min_{\sigma:\sigma_0=A,\sigma_N=B} \rho(\sigma) = \min_{\sigma:\sigma_0=A,\sigma_N=B} \sum_n \Delta\rho_n(\sigma_n,\sigma_{n+1}) \tag{1.25}$$

This algorithm is illustrated in figure 1.10 and in equation 1.26 shows the following example: We are looking for the optimal path from A to B, and we have computed the smallest metrics to get to the points one step from the destination B, points C and D. The least value of overall metric to get to B will be chosen as a minimum of metric needed to get to C with added metric to get from C to B or equivalently with D. Whichever one of these paths will have smaller overall metric is chosen as the ideal path.

$$\rho_{\text{min}}(A,B) = \min\left(\rho_{\text{min}}(A,C) + \Delta\rho(C,B), \quad \rho_{\text{min}}(A,D) + \Delta\rho(D,B)\right) \tag{1.26}$$

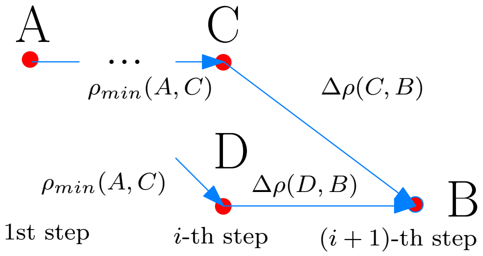


**Figure 1.10:** Example of Viterbi algorithm

## 1.7 Estimation theory

In the previous sections, the fundamentals of digital communication were established. Now a basic introduction to estimation and detection theory will happen.

Estimation theory focuses on estimating the actual value of unknown parameters $\theta$. The parameters can either be deterministic or random with a known or an unknown PDF. For the estimation, we will need a mathematical model of observation, in our case, a conditional PDF.

For example let us assume an AWGN channel with variance $\sigma^2$, BPSK modulation and the unknown parameter $\theta$ is the transmitted binary data. The observation model $p(\boldsymbol{x}|\theta)$ is in equation 1.27 and is plotted in figure 1.11.

$$\begin{aligned} p(\boldsymbol{x}|\theta = 0) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[ -\frac{1}{2\sigma^2} \parallel \boldsymbol{x} + 1 \parallel^2 \right] \\ p(\boldsymbol{x}|\theta = 1) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[ -\frac{1}{2\sigma^2} \parallel \boldsymbol{x} - 1 \parallel^2 \right] \end{aligned} \tag{1.27}$$

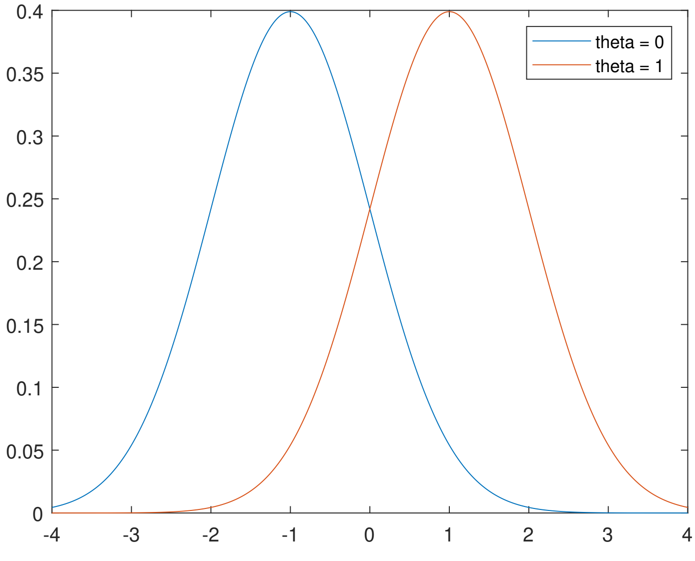


**Figure 1.11:** Different PDFs for different hypohesis

We can see that we get multiple shifted PDFs, each one for a different value of a parameter. For estimating the value of parameter from the received observation $\boldsymbol{x}$ we simply choose the parameter with the highest value of PDF for given value of observation $\boldsymbol{x}$.

### 1.7.1 Cramer-Rao Lower Bound

An important performance parameter of an estimator is its variance and how close it approaches an *ideal* estimator variance for a given observation model. The Cramer-Rao Lower Bound (CRLB for short) places a lower bound on the variance of an *unbiased* estimator (an estimator that, on average, yields the true value of the unknown parameter). The CRLB gives us the best possible variance and is a good comparison for the estimators. If the *regularity condition* 1.28 of the observation model is satisfied then the CRLB for that model is defined as 1.29.

$$\mathrm{E}\left[\frac{\partial \ln p(\boldsymbol{x}|\theta)}{\partial \theta}\right] = 0 \tag{1.28}$$

$$\operatorname{var}[\hat{\theta}] \geq \left(-\mathrm{E}\left[\frac{\partial^2 \ln p(\boldsymbol{x}|\theta)}{\partial \theta^2}\right]\right)^{-1} = \left(\mathrm{E}\left[\left(\frac{\partial \ln p(\boldsymbol{x}|\theta)}{\partial \theta}\right)^2\right]\right)^{-1} \tag{1.29}$$

**Proof:** Let us have an observation model $p(\boldsymbol{x})$ that satisfies regularity condition. Meaning the order of the derivation and integration is interchangeable as shown in 1.30. In the equation, a relation for derivative of a logarithm of a function was used $\frac{\partial \ln p(\boldsymbol{x}|\theta)}{\partial \theta} = \frac{1}{p(\boldsymbol{x}|\theta)} \frac{\partial p(\boldsymbol{x}|\theta)}{\partial \theta}$.

$$\int \frac{\partial \ln p(\boldsymbol{x}|\theta)}{\partial \theta} p(\boldsymbol{x}|\theta)\mathrm{d}\boldsymbol{x} = \int \frac{\partial p(\boldsymbol{x}|\theta)}{\partial \theta}\mathrm{d}\boldsymbol{x} \overset{\text{reg.}}{=} \frac{\partial}{\partial \theta}\int p(\boldsymbol{x}|\theta)\mathrm{d}\boldsymbol{x} = 0 \tag{1.30}$$

The relation for bias $b$ is described in equation 1.31. $\theta$ is the true value of a parameter and $\hat{\theta}$ is its estimate.

$$b = \mathrm{E}\left[\hat{\theta} - \theta\right] = \int (\hat{\theta} - \theta)p(\boldsymbol{x}|\theta)\mathrm{d}\boldsymbol{x} \tag{1.31}$$

The derivation of bias $b$ from is shown in 1.32.

$$\begin{aligned}\frac{\partial b}{\partial \theta} &= \frac{\partial}{\partial \theta}\int (\hat{\theta} - \theta)p(\boldsymbol{x}|\theta)\mathrm{d}\boldsymbol{x} \\ &= \int \frac{\partial(\hat{\theta} - \theta)}{\partial \theta} p(\boldsymbol{x}|\theta)\mathrm{d}\boldsymbol{x} + \int (\hat{\theta} - \theta)\frac{\partial p(\boldsymbol{x}|\theta)}{\partial \theta}\mathrm{d}\boldsymbol{x} \\ &= -\int p(\boldsymbol{x}|\theta)\mathrm{d}\boldsymbol{x} + \int (\hat{\theta} - \theta)\frac{\partial \ln p(\boldsymbol{x}|\theta)}{\partial \theta} p(\boldsymbol{x}|\theta)\mathrm{d}\boldsymbol{x}\end{aligned} \tag{1.32}$$

If the estimator is unbiased, then the derivative $\frac{\partial b}{\partial \theta}$ is equal to zero, and then both integrals are equal to 1 1.33.

$$1 = \int (\hat{\theta} - \theta)\frac{\partial \ln p(\boldsymbol{x}|\theta)}{\partial \theta} p(\boldsymbol{x}|\theta)\mathrm{d}\boldsymbol{x} \tag{1.33}$$

Now using the Cauchy-Schwarz inequality $|\langle a, b\rangle|^2 \leq \parallel a \parallel^2 \parallel b \parallel^2$ applied to the integral in 1.33, we get 1.34 which can be rewritten into the CRLB definition in 1.35.

$$\begin{aligned} 1^2 &\leq \left(\int (\hat{\theta}-\theta)^2 p(\boldsymbol{x}|\theta)\mathrm{d}\boldsymbol{x}\right)\left(\int \left(\frac{\partial \ln p(\boldsymbol{x}|\theta)}{\partial \theta}\right)^2 p(\boldsymbol{x}|\theta)\mathrm{d}\boldsymbol{x}\right) \\ &\leq \operatorname{var}[\hat{\theta}] \cdot \mathrm{E}\left[\left(\frac{\partial \ln p(\boldsymbol{x}|\theta)}{\partial \theta}\right)^2\right] \end{aligned} \tag{1.34}$$

$$\operatorname{var}[\hat{\theta}] \geq \left(\mathrm{E}\left[\left(\frac{\partial \ln p(\boldsymbol{x}|\theta)}{\partial \theta}\right)^2\right]\right)^{-1} \tag{1.35}$$

I have taken the equations and the proof from the [6, presentation 01].

### 1.7.2 Maximum-Likelihood Estimator

I will now describe the most used type of estimator based on the maximum likelihood principle. This estimator finds the value of an unknown parameter $\theta$ as a value that maximizes the likelihood function for one given observation $\boldsymbol{x}$ over all possible initial parameter values $\check{\theta}$ as shown in equation 1.36 and in figure 1.12.

$$\hat{\theta} = \max_{\check{\theta}} p(\boldsymbol{x}|\check{\theta}) \tag{1.36}$$

In general, the ML estimator has the properties of being unbiased. Another great property of the ML estimator is that for large data records, the distribution function of the estimation approaches the Gaussian curve with a mean equal to the true value of parameter $\theta$. Its variance is equal to the CRLB for the given observation model. We say that ML estimator is *asymptotically optimal* [3, p. 157].

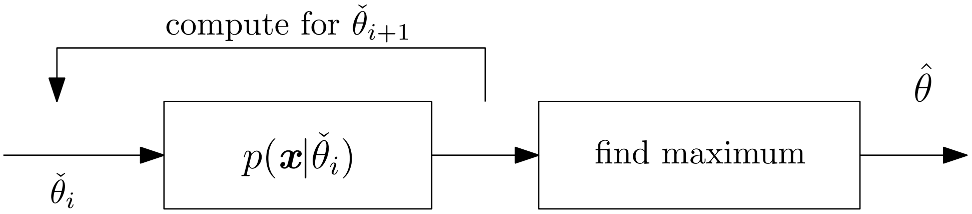

compute for $\check{\theta}_{i+1}$
$\hat{\theta}$
$p(\boldsymbol{x}|\check{\theta}_i)$
find maximum
$\check{\theta}_i$

**Figure 1.12:** ML estimator scheme

## 1.8 Detection Theory

In the previous chapter, we talked about the theory of estimating parameters. Now, I will lightly introduce the basics of signal detection theory.

The signal detector has received an observation $\boldsymbol{x}$ and his goal is to find out whether a useful signal containing information is in that received observation or not.

The detector will be using hypothesis testing to determine whether a useful signal is present in the observation. For binary detection, there will be two types of signals - null or alternative signal 1.37. If the signal detector denies the alternative hypothesis, it automatically assumes the null signal is present in the received signal and vice-versa.

$$\boldsymbol{s} = \begin{cases} \boldsymbol{s}^{(0)} & = \boldsymbol{0}, \quad \text{null signal} \\ \boldsymbol{s}^{(1)} & \neq \boldsymbol{0}, \quad \text{alternative signal} \end{cases} \tag{1.37}$$

In relation to the null/alternative signals, we can define the following probabilities:

- Detection probability $P_D = P(1|1) = \Pr\{\boldsymbol{s} = \boldsymbol{s}^{(1)} | \boldsymbol{s} = \boldsymbol{s}^{(1)}\}$
- Miss probability $P_M = P(0|1) = \Pr\{\boldsymbol{s} = \boldsymbol{s}^{(0)} | \boldsymbol{s} = \boldsymbol{s}^{(1)}\}$
- False Alarm probability $P_{FA} = P(1|0) = \Pr\{\boldsymbol{s} = \boldsymbol{s}^{(1)} | \boldsymbol{s} = \boldsymbol{s}^{(0)}\}$
- Null-detection probability $P_0 = P(0|0) = \Pr\{\boldsymbol{s} = \boldsymbol{s}^{(0)} | \boldsymbol{s} = \boldsymbol{s}^{(0)}\}$

### 1.8.1 Neyman-Pearson Theorem

Neyman-Pearson theorem is a binary hypothesis detector that maximizes the detection probability $P_D$ for a given false alarm probability $P_{FA} = \alpha$ if the decision is the likelihood ratio test. If the likelihood ratio function $\Lambda(\boldsymbol{x})$ 1.39 is bigger than a certain *threshold* $\gamma$ related to $\alpha$ in 1.40, the detector assumes the received signal has a useful signal in it. Otherwise, it assumes a null signal has been received 1.38.

$$\boldsymbol{s} = \begin{cases} \boldsymbol{s}^{(0)} & \text{for } \Lambda(\boldsymbol{x}) \leq \gamma \\ \boldsymbol{s}^{(1)} & \text{for } \Lambda(\boldsymbol{x}) > \gamma \end{cases} \tag{1.38}$$

$$\Lambda(\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\boldsymbol{s}^{(1)})}{p(\boldsymbol{x}|\boldsymbol{s}^{(0)})} \tag{1.39}$$

$$\alpha = \int_{\boldsymbol{x}:\Lambda(\boldsymbol{x})>\gamma} p(\boldsymbol{x}|\boldsymbol{s}^{(0)}) \mathrm{d}\boldsymbol{x} \tag{1.40}$$

Definitions are taken from [4, p. 61 - 65] and [6, presentation 05].

## 1.9 Introduction to WPNC

This section will establish the needed fundamentals of Wireless Physical layer Network Coding.

The WPNC introduces a new idea that could replace the classic point-to-multipoint communication in the future where the receivers are *network-aware*. This receiver knows the network's structure as a whole and is aware of the possible signals this receiver may receive. As we will see, it is favorable for the receiver to receive more non-orthogonal signals at once to increase the overall throughput of the network.

### 1.9.1 2-Way Relay Channel

Let us take the following network structure 1.13 as an example. This structure is called the *Two-Way Relay Channel* (2WRC) and is the most straightforward network that can utilize the WPNC. We have two sources, A and B, that wish to communicate, but their signals can not reach the other source. The communication is conveyed with the help of the relay node R.

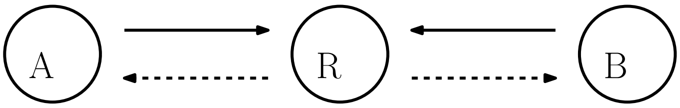


**Figure 1.13:** Two-Way Relay Channel topology

In traditional communication using the time division, the communication would require four time slots to complete. However, using the WPNC, the communication will require just two time slots 1.14. As we will see, the relay node will utilize the *network coding* approach. The node will calculate the functions of the received data from both sources and transmits the information containing both the data streams. With the knowledge of the data the sources transmitted, the sources can decode the information aimed at them.

### 1.9.2 Hierarchical decode and forward

Now let us focus on the relay node's strategy when the communication happens. The strategy is called *hierarchical decode and forward* (HDF), and its basics were lightly introduced in the previous section.

The HDF will use the mapping function directly from the overlapped received signal from both sources. The mapping function for binary transmitted data will be a simple exclusive or operation (XOR) [2, p. 20, 21].

After the relay node has mapped the data, the node is ready to transmit the data back to the sources. The sources with the knowledge of their data can decode the information from the source by applying another XOR function to the received data 1.41.

$$(s_A \otimes s_B) \otimes s_B = s_A \tag{1.41}$$

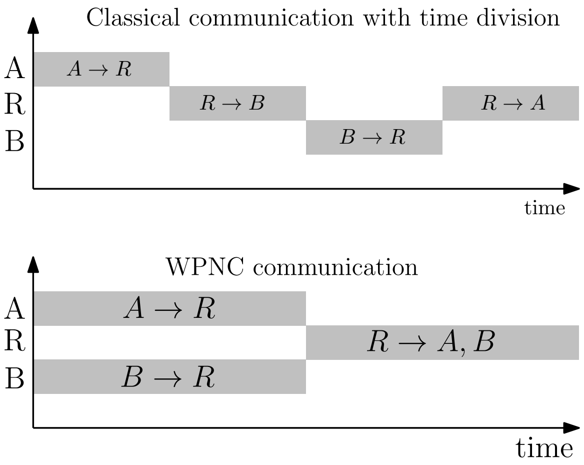


**Figure 1.14:** Comparison between TDMA and WPNC

### 1.9.3 H-constellation

As described previously, a mapping function exists at the relay node that turns the received signal into binary data. The mapping function will be a simple minimum distance detector as described in 1.5.3. However, the points that will define the decision region will not be a traditional, already defined constellation.

Because the received signal consists of multiple separate signals from sources, the constellation the signals will create will be a combination of multiple constellations. This constellation will be called the *Hierarchical constellation* (H-constellation).

Assume that both the sources used a simple BPSK modulation as defined in 1.4. In this case, the H-constellation is a combination of two BPSK constellations as shown in 1.15. We can see that the resulting constellation consists of three points: one for both sources transmitting 0, one for both sources transmitting 1, and 2 points representing the situation where the sources transmitted opposite bits.

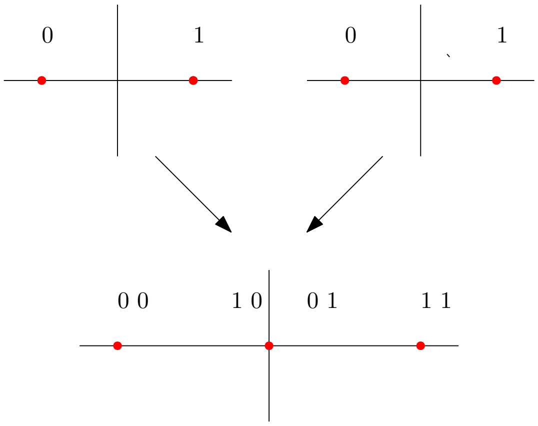


**Figure 1.15:** H-constellation for two BPSK modulations

When we know how the H-constellation will look, we can imagine the mapping function at the relay node. If the received signal is closer to the middle where there are the two overlapped constellation points, the node assumes the sources transmitted the opposite bits, and the result of the mapping function (a XOR function in our case) is 1. In another case, the result is 0 as both sources transmitted the same bits.

## Relative fading

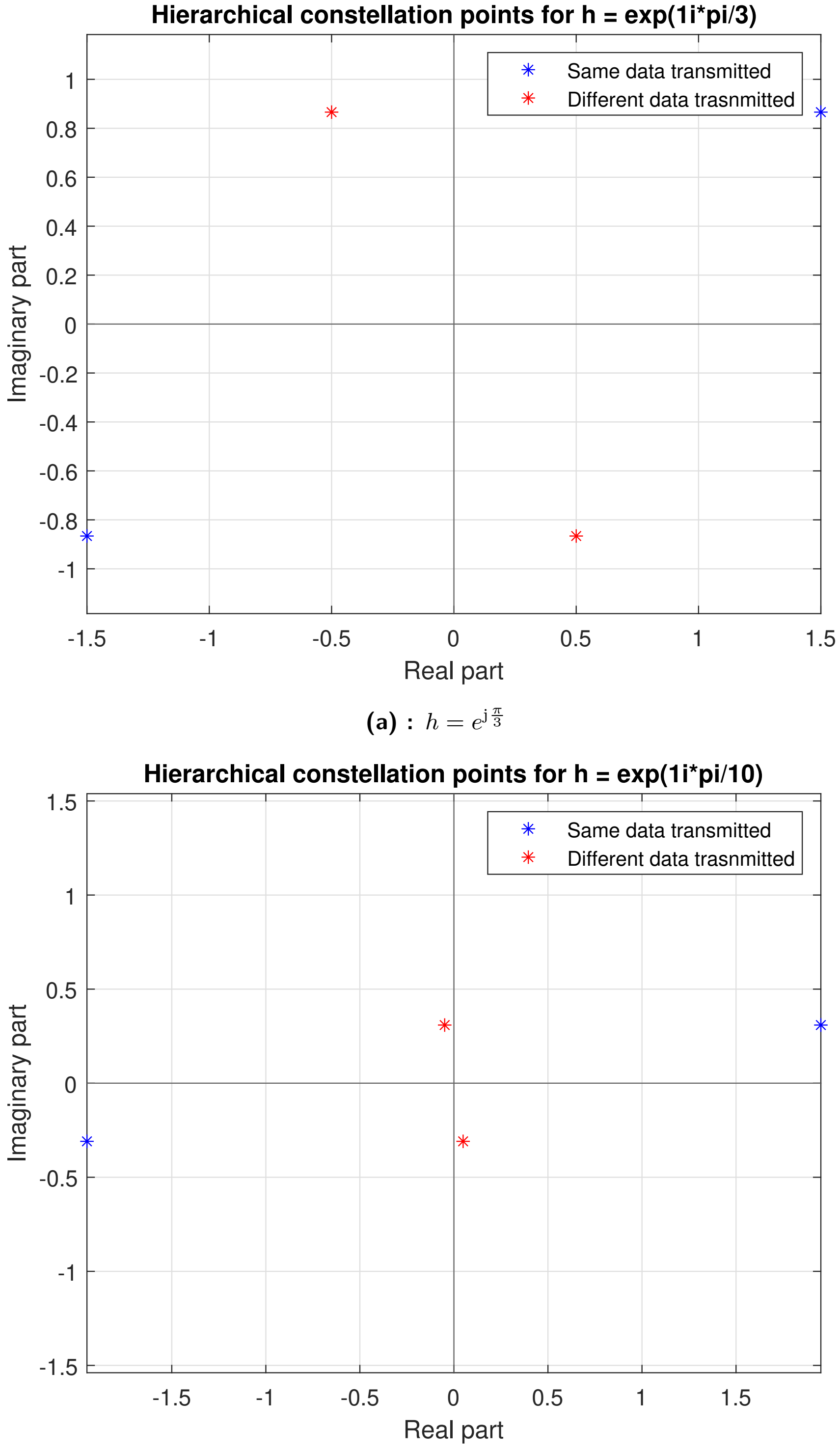
In reality, we need to consider the effects of the channel, which will also affect how the H-constellation will look. The exact parameter that will significantly affect the H-constellation is a phase shift.

We have received two signals, $s_A$, and $s_B$. Each has passed through the channel, and each useful signal has been affected by a fading $h_A$ and $h_B$. The node has received the signal x combined with both these noisy signals 1.42.

$$x = h_A s_A + h_B s_B = h_A(s_A + \frac{h_B}{h_A} s_B) \tag{1.42}$$

We can negate the effects of the *singular fading* $h_A$ by zero-forcing so this parameter plays no role in the way the H-constellation will look. What plays role is the effect of the *relative fading* $h = \frac{h_B}{h_A}$ [2, p. 31, 32]. The relative fading represents a relative phase shift of one signal compared to the other. In constellation space the channel symbols will also be relatively shifted.

We can take a look how the H-constellation will look for BPSK modulations with various values of $h$ in figures 1.16.



**(a) :** $h = e^{\mathrm{j}\frac{\pi}{3}}$

**(b) :** $h = e^{\mathrm{j}\frac{\pi}{10}}$

**Figure 1.16:** H-constellation for BPSK modulation for different values of $h$

# Chapter 2

# WPNC in 2WRC with realistic channel model

In this chapter, I will talk more in-depth about one particular example where I will include all the previous theoretical parts implemented for one such example. This case will later be implemented in Matlab.

## 2.1 General overview

I will begin by describing the digital communication scheme, including all the function blocks described more deeply in later sections. Let us begin.

The digital communication will proceed between two communication nodes A and B in a simple topology *Two-Way Relay Channel* as shown in 1.13. Both the sources will transmit *coded* binary stream information using linear PSK modulations to the relay channel with the CAZAC sequence as a pilot to determine random channel parameters in processing at the receiver's end. Each node will have its unique CAZAC sequence with its unique prime $M, N$ parameters. The CAZAC sequence is defined in equation 2.1.

$$\text{CAZAC}(M, N) = \left[ \exp\left( \mathrm{j}\pi \frac{M}{N} i(i+1) \right) \right]_{i=0}^{i=N-1} \tag{2.1}$$

Both the signals will pass through the LTI AWGN channel with random nuisance parameters 1.6. The parameters are following:

- Random phase shift with uniform PDF $\varphi \in \mathcal{U}(0, 2\pi)$
- Random delay equal to multiples of symbol period $N_s$, $\tau = n \cdot N_s, \quad n \in \mathbb{N}$
- Complex AWGN dependent on chosen SNR

The relay node will then process the information from both sources, knowing that the signals might not be perfectly synchronized. The imperfection of the synchronization will take effect on the output signal.

Lastly, the noisy signal from the relay node will be detected at the sources themselves using the matched filter. With the knowledge of all the CAZAC pilots in the system and its transmitted data, the data from the other source will be detected and decoded.

## 2.2 Coding

The coding of binary data streams $\boldsymbol{d}_A, \boldsymbol{d}_B$ will be done using convolution coding with matrix $\boldsymbol{G}$ in equation 2.2. The relation between encoded and message bits is in equation 2.3.

$$\boldsymbol{G} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \tag{2.2}$$

$$\begin{aligned} c_{n,1} &= d_{n-1} + d_n \\ c_{n,1} &= d_n \end{aligned} \tag{2.3}$$

Lastly the function $\mod 2$ will be used on coded sequence so the output is a binary sequence. For a message $\boldsymbol{d}$ of length $N_d$ we get coded sequence $\boldsymbol{c}$ of length $2(N_d - 1)$.

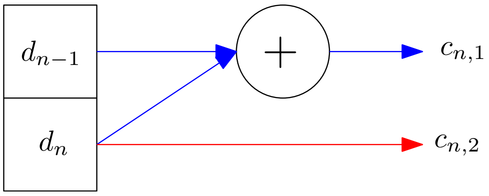


**Figure 2.1:** Implemented code

## 2.3 Signal modulation

All nodes will be using the same linear memory-less modulation that uses phase-shift keying, namely BPSK or QPSK. The coded data in the form of a binary stream $\boldsymbol{c}$ will be transformed into channel symbol stream $\boldsymbol{q}$ depending on the used modulation. Then the channel symbols will be up-sampled with chosen symbol period.

Lastly, the up-sampled channel symbol stream will pass through a filter with an impulse response equal to the RRC pulse. The output of the modulator is the continuous-time signal $s(t)$ that can be detected at the receiver's end using matched filter.

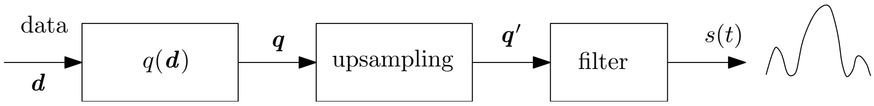


**Figure 2.2:** Scheme of the data modulation

## 2.4 Communication channel

Then, the signal passes through the parametric channel where it is affected by random phase shift with uniform distribution, random time delay equal to integers of symbol period and complex AWGN related to chosen SNR $\gamma$.

The relation for SNR is $\gamma = \frac{\epsilon_b}{N_0}$ [1, p. 176] where $\epsilon_b$ is signal energy per bit and $\frac{N_0}{2}$ is a power spectral density of the noise. The relation between $N_0$ and noise variance $\sigma_w$ is $\sigma_w = \sqrt{2N_0}$.

With that the relation between SNR and noise variance is: $\gamma = \frac{2\epsilon_b}{\sigma_w^2}$. We will be using SNR in dB. With that the relation is:

$$\gamma = 10\log_{10}\left(\frac{2\epsilon_b}{\sigma_w^2}\right) \rightarrow \sigma_w^2 = 10^{-\frac{\gamma}{10}}2\epsilon_b = 10^{-\frac{\gamma}{10}}\frac{2\epsilon_s}{\log_2 M_d} \tag{2.4}$$

The $\epsilon_s$ is symbol energy equal to $\frac{1}{2}$ for both BPSK and QPSK modulations, and $M_d$ is the cardinality of the alphabet. The BPSK has cardinality $M_d = 2$ and QPSK has $M_d = 4$. For these modulations, the relation between noise variance and SNR is shown in equations 2.5.

$$\begin{aligned} \text{BPSK}: &\quad \sigma_w^2 = 10^{-\frac{\gamma}{10}} \\ \text{QPSK}: &\quad \sigma_w^2 = \frac{1}{2}10^{-\frac{\gamma}{10}} \end{aligned} \tag{2.5}$$

## 2.5 Relay node processing

The relay node receives the overlapped, not synchronized mesh of both signals, and the node needs to determine whether what it received has anything useful from either of the sources.

If the relay node determines that it is receiving a signal containing information, the node needs to determine the value of channel parameters that affected the signal. In our case, these parameters are *phase shift and delay*. Lastly, with the knowledge of the parameters, the relay node can detect the received data and forward it to the sources themselves. The general scheme is shown in figure 2.3.

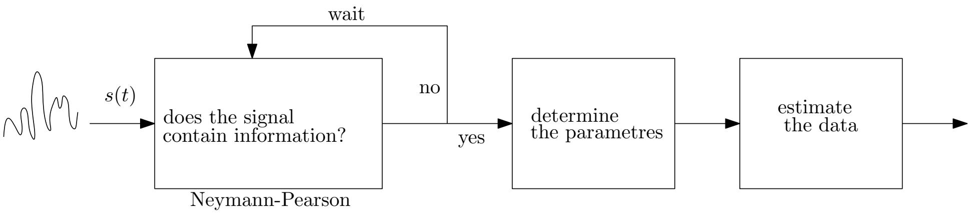


**Figure 2.3:** General scheme for relay node processing

### 2.5.1 Neyman-Pearson Theorem

Using the Neyman-Pearson theorem, the node decides whether we have received the useful signal or purely noise containing no information. In the following equations, I will show the threshold value for accepting the alternative hypothesis for a given false alarm probability $\alpha$.

Let us have a relation between received signal and sent signal $\boldsymbol{x} = \boldsymbol{s} + \boldsymbol{w}$ where $\boldsymbol{w}$ is AWGN with variance $\sigma_w^2$. Let us have null $H_0$ and alternative hypothesis $H_1$ as shown in equations 2.6.

$$\begin{aligned} H_0 : y &= |\langle \boldsymbol{0} + \boldsymbol{w}, \boldsymbol{s}\rangle|^2 = |\xi|^2 \\ H_1 : y &= |\langle \boldsymbol{s} + \boldsymbol{w}, \boldsymbol{s}\rangle|^2 = |\xi + \epsilon_s|^2 \end{aligned} \tag{2.6}$$

The $\epsilon_s$ denotes the energy of signal $\boldsymbol{s}$. The $\xi$ has *zero mean Gaussian* PDF with variance $\sigma_w^2 \epsilon_s$ as proven in 2.7.

$$\begin{aligned} \text{var}[\xi] = \text{E}[|\xi|^2] = \text{E}[|\langle \boldsymbol{w}, \boldsymbol{s}\rangle|^2] &= \text{E}[\sum_i \sum_j w_i s_i^* w_j^* s_j] \\ &= \sum_i \sum_j \text{E}[w_i w_j^*] s_i^* s_j = \sum_i \sigma_w^2 s_i^* s_i = \sigma_w^2 \epsilon_s^2 = \sigma_\xi^2 \end{aligned} \tag{2.7}$$

Now let's take a look at PDF of the absolute value of cross-energy $y$. For simplification substitution $y' = \frac{2y}{\sigma_\xi^2}$ is used. The PDF will be $\chi^2$, for null hypothesis the PDF will be centralized, for alternative hypothesis it will be decentralized with the center $\lambda = \frac{2\epsilon_s^2}{\sigma_\xi^2}$. Both the PDFs in equation 2.8 are with $k = 2$ degrees of freedom. The $I_0(\cdot)$ represents modified Bessel function.

$$\begin{aligned} p(y'|0) &= \frac{1}{2}e^{-\frac{y'}{2}} \\ p(y'|1) &= \frac{1}{2}e^{-\frac{y'+\lambda}{2}} I_0(\sqrt{\lambda y'}) \end{aligned} \tag{2.8}$$

The likelihood function defined in 1.39 will be:

$$\Lambda = \frac{p(y'|1)}{p(y'|0)} = e^{-\frac{\lambda}{2}} I_0(\sqrt{y'\lambda}) \tag{2.9}$$

The signal detector will decide to accept the alternative hypothesis if the likelihood function $\Lambda$ is bigger than some arbitrarily chosen value $\gamma$ 1.40.

$$e^{-\frac{\lambda}{2}} I_0(\sqrt{y'\lambda}) > \gamma \Rightarrow y' > \frac{(I_0^{-1}(\exp\frac{\lambda}{2}\gamma))^2}{\lambda} = \beta \tag{2.10}$$

We can find the value of $\beta$ depending on chosen $\alpha$ as:

$$\alpha = \int_\beta^\infty p(y'|0)\mathrm{d}y' = e^{-\frac{-\beta}{2}} \Rightarrow \beta = 2\ln\frac{1}{\alpha} \tag{2.11}$$

Finally, we can write the condition under which the detector assumes the useful signal is present in the received signal in 2.12.

$$\frac{2y}{\sigma_w^2\epsilon_s^2} > 2\ln\frac{1}{\alpha} \Rightarrow y > \sigma_w^2\epsilon_s^2\ln\frac{1}{\alpha} \tag{2.12}$$

### 2.5.2 Estimating the channel parameters

The useful signal received some changes by passing through the communication channel. In order to find out the information modulated in the signal, we need to *estimate* phase shift and delay. For the estimation, we will be using the known signal pilot in the form of the CAZAC sequence defined in equation 2.1.

We will use the Joint Maximum Likelihood estimator for the estimation. Firstly the delay will be estimated, and assuming the estimation is correct, the phase shift can also be estimated. The delay will be estimated using the Maximum Likelihood principle 1.12 as is shown in 2.13.

$$\hat{\tau} = \arg\max_{\check{\tau}} |\langle x(t), s(t+\check{\tau})\rangle| \tag{2.13}$$

The $x(t)$ is the received noisy signal and $s(t+\tau)$ is the known signal CAZAC sequence shifted in the time domain by an integer of symbol period

$\tau$. We are searching the maximum of the absolute value cross-energy of the received signal $x(t)$ and the sequence we are searching in that signal.

With the knowledge of time delay, the estimator can estimate the phase shift the channel has done to the useful signal. We can see that the phase shift will be an angle of the maximum of the cross-correlation function from the previous equation. Mathematically the relation is written in 2.14.

$$\hat{\varphi} = \angle\langle x(t), s(t+\hat{\tau})\rangle \tag{2.14}$$

Where again the $x(t)$ is the received noisy signal and $s(t+\hat{\tau})$ is time-shifted known CAZAC sequence modulated onto pulses. The time shift $\hat{\tau}$ is equal to the estimated time delay the joint estimator gave us in the previous step.

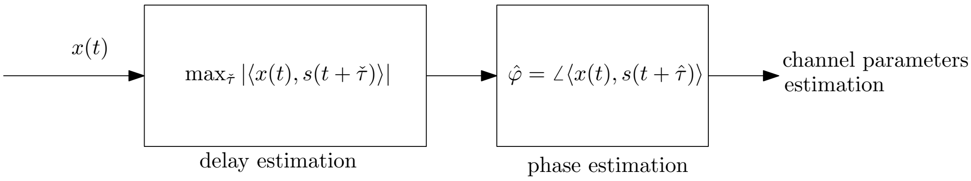

$x(t)$
$\max_{\check{\tau}} |\langle x(t), s(t+\check{\tau})\rangle|$
$\hat{\varphi} = \angle\langle x(t), s(t+\hat{\tau})\rangle$
channel parameters estimation
delay estimation
phase estimation

**Figure 2.4:** Scheme of Joint ML estimator

## Cramer-Rao Lower Bound

In the chapter where the numerical results done using Matlab will be presented, I will compare the variance of the phase estimator to the Cramer-Rao Lower Bound for this specific case. Here for simplicity, I will show the computation of the CRLB for the case where only a single signal is transmitted. For the case of both signals transmitting, I will refer to the pages in one reference of this work where the computation is done.

For a single source transmitting the observation model is given by:

$$p(\boldsymbol{x}|\varphi) = \frac{1}{\pi^N \sigma_w^2} \exp -\frac{1}{\sigma_w^2} \parallel \boldsymbol{x} - e^{\mathrm{j}\varphi}\boldsymbol{s} \parallel \tag{2.15}$$

If we want to compute the CRLB we need to make sure the *regularity condition* 1.28 holds true. By writing each components of the inner product $\parallel \boldsymbol{x} - e^{\mathrm{j}\varphi}\boldsymbol{s} \parallel$ we get equation 2.16.

$$\frac{\partial \ln p(\boldsymbol{x}|\varphi)}{\partial \varphi} = -\frac{1}{\sigma_w^2}\left(-\mathrm{j}e^{\mathrm{j}\varphi}\langle \boldsymbol{s}, \boldsymbol{x}\rangle + \mathrm{j}e^{-\mathrm{j}\varphi}\langle \boldsymbol{x}, \boldsymbol{s}\rangle\right) \tag{2.16}$$

The regularity condition will hold for the phase shift uniformly distributed between 0 and $2\pi$. We will integrate the complex exponential over its period, and nothing else depends on the $\varphi$, thus resulting in 0 mean value.

We can now compute the CRLB. For that, we will need a second derivative of the logarithm function.

$$\frac{\partial^2 \ln p(\boldsymbol{x}|\varphi)}{\partial \varphi^2} = -\frac{1}{\sigma_w^2}\left(e^{\mathrm{j}\varphi}\langle \boldsymbol{s}, \boldsymbol{x}\rangle + e^{-\mathrm{j}\varphi}\langle \boldsymbol{x}, \boldsymbol{s}\rangle\right) \tag{2.17}$$

Now all we need is the computation of the average over the $\varphi$ distribution. We will use the relation for the observation $\boldsymbol{x} = e^{\mathrm{j}\varphi}\boldsymbol{s} + \boldsymbol{w}$ where $\boldsymbol{w}$ is zero-mean AWGN.

$$\begin{aligned} e^{\mathrm{j}\varphi}\langle \boldsymbol{s}, \boldsymbol{x}\rangle + e^{-\mathrm{j}\varphi}\langle \boldsymbol{x}, \boldsymbol{s}\rangle &= e^{\mathrm{j}\varphi}(\langle \boldsymbol{s}, \boldsymbol{s}\rangle e^{-\mathrm{j}\varphi} + \langle \boldsymbol{s}, \boldsymbol{w}\rangle) + e^{-\mathrm{j}\varphi}(e^{+\mathrm{j}\varphi}\langle \boldsymbol{s}, \boldsymbol{s}\rangle + \langle \boldsymbol{s}, \boldsymbol{w}\rangle) \\ &= 2 \parallel \boldsymbol{s} \parallel^2 + 2\mathrm{Re}\big[\langle \boldsymbol{s}, \boldsymbol{w}\rangle\big] \end{aligned} \tag{2.18}$$

Using the fact that the AWGN has zero mean we get:

$$\mathrm{E}\left[-\frac{1}{\sigma_w^2}\Big(2 \parallel \boldsymbol{s} \parallel^2 + 2\mathrm{Re}\big[\langle \boldsymbol{s}, \boldsymbol{w}\rangle\big)\big]\right] = -\frac{2 \parallel \boldsymbol{s} \parallel^2}{\sigma_w^2} \tag{2.19}$$

Thus the lower bound for the variance of the estimator is:

$$\mathrm{var}[\check{\varphi}] \geq \frac{\sigma_w^2}{2 \parallel \boldsymbol{s} \parallel^2} \tag{2.20}$$

The computation of the variance lower bound for case with both signals transmitting is more complex and is done in [2, p. 210 - 212]. For this case the observation model is:

$$p(\boldsymbol{x}|\varphi_A, \varphi_B) = \frac{1}{\pi^N \sigma_w^2} \exp - \frac{1}{\sigma_w^2} \parallel \boldsymbol{x} - e^{\mathrm{j}\varphi_A}\boldsymbol{s}_A - e^{\mathrm{j}\varphi_B}\boldsymbol{s}_B \parallel \tag{2.21}$$

And the lower bound for the variances will be:

$$\begin{aligned} \mathrm{var}[\check{\varphi_A}] &\geq \frac{\sigma_w^2}{2 \parallel \boldsymbol{s}_A \parallel^2 \Big(1 - \mathrm{Re}\big[e^{\mathrm{j}(\check{\varphi_A} - \check{\varphi_B})}\eta\big]^2\Big)} \\ \mathrm{var}[\check{\varphi_B}] &\geq \frac{\sigma_w^2}{2 \parallel \boldsymbol{s}_B \parallel^2 \Big(1 - \mathrm{Re}\big[e^{\mathrm{j}(\check{\varphi_A} - \check{\varphi_B})}\eta\big]^2\Big)} \end{aligned} \tag{2.22}$$

The $\eta$ denotes scaled inner product of the two signals.

$$\eta = \frac{\langle \boldsymbol{s}_A, \boldsymbol{s}_B\rangle}{\parallel \boldsymbol{s}_A \parallel \parallel \boldsymbol{s}_B \parallel} \tag{2.23}$$

### 2.5.3 Data detecting

#### Single signal case

If only a single signal is transmitted, the relay node has a straightforward goal. All the node needs to do is detect the data from that source and forward it in a later step. Firstly the node will use zero-forcing of the phase shift on the received signal, so the effect of the shift is negated. Then using matched filter and the decision regions in the constellation space, the node can detect the data.

### Multiple signals case

If multiple sources transmitted, the node estimates all the necessary channel parameters. After this has been done, the node will use zero-forcing of two nuisances:

1. The phase shift zero-forcing of the source that had transmitted earlier
2. The CAZAC pilot of the later signal as it is no longer needed after the estimation of the parameters

After the zero-forcing has happened, we can use matched filter to give us received constellation points and then use the decision regions to detect the data. For detecting the non-overlapped data, the decision regions are given by used modulation (BPSK or QPSK), possibly by relative fading $h$.

To detect the overlapped data, the received constellation points will be compared with the H-constellation points generated for specific modulation and the specific relative fading $h$.

The figure 2.5 illustrates the scheme of data detection at the relay node for both cases.

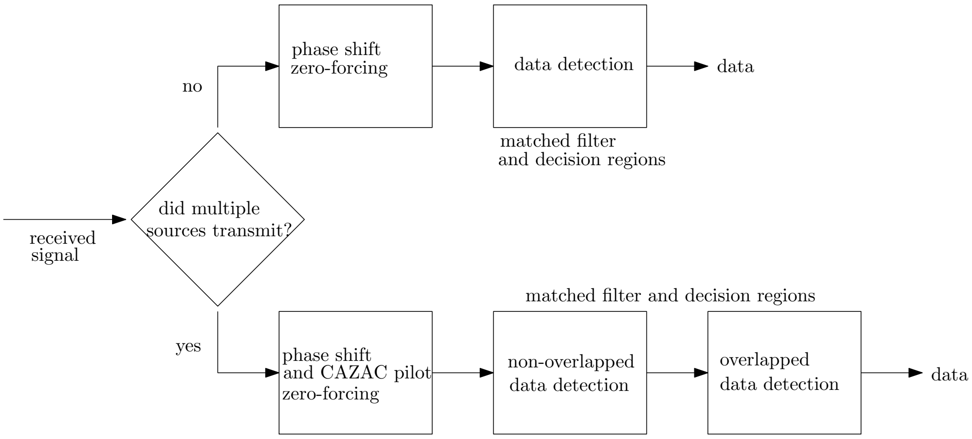

phase shift zero-forcing
data detection
data
no
matched filter and decision regions
received signal
did multiple sources transmit?
matched filter and decision regions
yes
phase shift and CAZAC pilot zero-forcing
non-overlapped data detection
overlapped data detection
data

**Figure 2.5:** Scheme of data detection on relay node

## 2.5.4 Broadcasting the detected data

### Single signal case

If the relay node assumed using the Neyman-Pearson theorem that only one source transmitted data, the node would act as a forwarder of the received data. To signify that only one signal was transmitted, the forwarded signal's pilot signal will be the same as the pilot the transmitting source used. Meaning if only source A was broadcasting, the forwarded message will start with the pilot belonging to source A with its characteristic $M, N$ constants in the CAZAC sequence. After the CAZAC pilot, the binary detected data sequence modulated onto the RRC pulse follows.

### Multiple signals case

If both signals have been determined to be present at the relay node, the broadcasted signal will have to contain additional information along with the detected data itself. With that, the broadcasted payload will have three parts.

1. Pilot signal in the form of CAZAC sequence belonging to the relay node
2. *Header* containing information on which source transmitted earlier and what is the relative delay between the two received signal
3. The xorred detected data itself

The first half of the bits assigned to the header is a stream of either all 0s or all 1s. 1s if A transmitted earlier and 0s if B transmitted earlier. The second half of the header is a binary number that represents the number of symbol periods the relative delay $\Delta\tau$ between two signals amounted. This is all the additional information the sources need along with the transmitted data itself.

The first part of the xorred data stream is the relay node's data from overlapped signals. The second half of the xorred data stream is the data that the node received individually. The node first detected the data, turned it into a binary stream, and then applied the xor function to these two streams of the same length. Effectively the data payload the relay node transmits is xorred detected data where the xor function has been applied to the circularly shifted data stream from the earlier source and the data stream from the later source.

The figure 2.6 illustrates the structure of the data payload from the relay node. Note that the pilots of both signals can overlap in the implementation. The relay node can process the signals if the relative delay $\Delta\tau$ is equal to any integer of symbol period.

After the data payload has been created, the same procedure with modulation and passing through the channel is done.

## 2.6 Processing at the sources

The end processing is straightforward. The sources receive a signal from the relay node. If the source assumes the message is aimed at it (in other words, the received message contains a CAZAC sequence other than the sequence belonging to that source), the sources will process the received signal. In this section, I will describe the processing of the signal using hard decision detection, the decoding of the message will be covered in the following section.

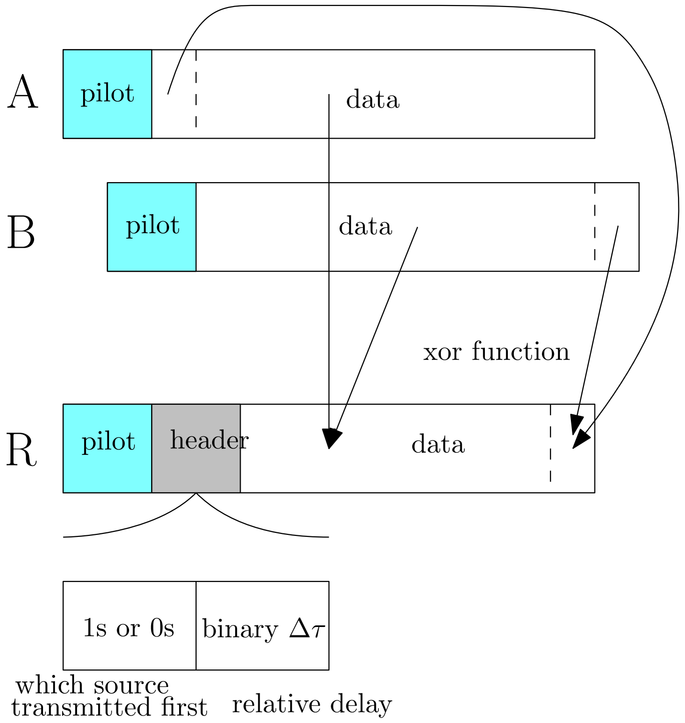


**Figure 2.6:** Data broadcasted by relay node if multiple sources transmitted

### Single signal case

If the signal contains the information only from the other source, the source does not need to dig out the information from the header. The source will only need to estimate the channel parameters, zero-force the phase shift, and using the matched filter the source can detect the data.

### Multiple signals case

If the signal is from both sources, or in other words, if the CAZAC pilot signal belongs to the relay node, the sources, after estimating the channel parameters, need to detect the data and get all the necessary information contained in the header.

With the knowledge of the relative delay, which source transmitted earlier and their transmitted data, the sources can reconstruct the data from another source. The signal from the xorred data payload has been created with the sequence of the earlier source being circularly shifted. The circular shift needs to be taken into account during the processing.

## 2.7 Decoding

The last part of the communication cycle is decoding the coded received sequence. Using the matched filter, we have constellation points corresponding to the signal the relay node has transmitted.

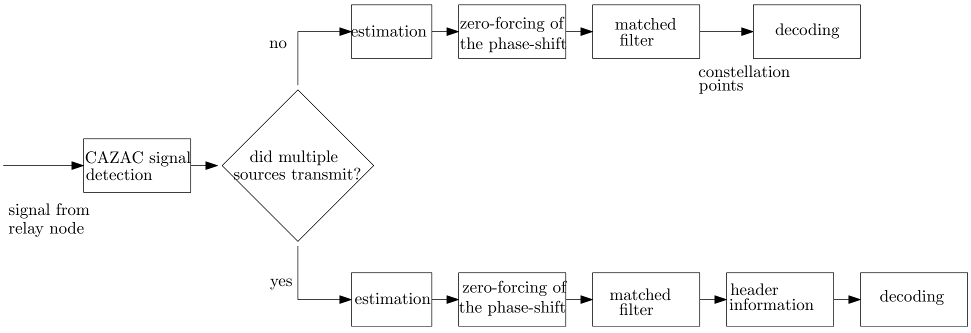


**Figure 2.7:** Processing at the sources

## 2.7.1 Single signal case

If only a single source transmitted, the relay node forwarded the message and added the appropriate pilot. When the source has received this message and is the intended destination, the source will apply a *Viterbi algorithm* with soft decision metric $\rho$ equal to the absolute distance in constellation space onto the constellation points the matched filter has given us. The algorithm returns an estimated data sequence that can be compared with the actual sequence the other source transmitted.

## 2.7.2 Multiple signals case

If multiple signals reached the relay node, the sources would need to detect information from the header using a hard decision metric. After getting the information regarding the relative delay and which source transmitted earlier, the sources will apply the inverse XOR function in constellation space and apply the Viterbi algorithm to the adjusted constellation space points.

### Applying inverse XOR function on received signal

After getting the information from the header, the source will apply the XOR function to the received constellation points. Using decision regions, the source will assume what the relay node transmitted and then adjusts the point to correspond with the assumed information the other source transmitted.

**Example**: Let us assume the received signal was modulated using QPSK modulation. Suppose the received message from the relay node in one symbol was 00, and the source transmitted 11 in that corresponding symbol. In that case, the source will shift the angle of the constellation point of that symbol by $\pi$ to apply an inverse XOR function in the constellation space. This phase shift makes the received constellation points represent the sequence the other source had transmitted.

The following tables 2.1 and 2.2 show the appropriate phase shifts $\varphi$ the sources need to apply to the received points $XOR$ if the source transmitted data $c$ in corresponding symbol and the new XOR symbol we get by applying said phase shift.

| $XOR\backslash C$ | **0** | **1** |
|---|---|---|
| **0** | $0,\ \varphi = 0$ | $1,\ \varphi = 0$ |
| **1** | $1,\ \varphi = \pi$ | $0,\ \varphi = \pi$ |

**Table 2.1:** XOR function in constellation space for BPSK modulation

| $XOR\backslash C$ | **00** | **01** | **11** | **10** |
|---|---|---|---|---|
| **00** | $00,\ \varphi = 0$ | $01,\ \varphi = \frac{\pi}{2}$ | $11,\ \varphi = \pi$ | $10,\ \varphi = -\frac{\pi}{2}$ |
| **01** | $01,\ \varphi = 0$ | $00,\ \varphi = -\frac{\pi}{2}$ | $10,\ \varphi = \pi$ | $11,\ \varphi = \frac{\pi}{2}$ |
| **11** | $11,\ \varphi = 0$ | $10,\ \varphi = -\frac{\pi}{2}$ | $00,\ \varphi = \pi$ | $01,\ \varphi = -\frac{\pi}{2}$ |
| **10** | $10,\ \varphi = 0$ | $11,\ \varphi = -\frac{\pi}{2}$ | $01,\ \varphi = \pi$ | $00,\ \varphi = \frac{\pi}{2}$ |

**Table 2.2:** XOR function in constellation space for QPSK modulation

### Viterbi algorithm

After applying the inverse XOR function to the constellation points and potentially inverse circular shift, the sources will apply a *Viterbi algorithm* to the constellation points. Using the algorithm, the source estimates the data sequence the other source had transmitted.

# Chapter 3

# Implementation

This chapter will describe the practical implementation of the communication system I have described in the previous chapter. I will describe the implementation of each function block in the attached files **`TWRC_BPSK_coded.m`** and **`TWRC_QPSK_coded.m`**. Both the Matlab scripts work by themselves and reader can follow this chapter along with the attached codes.

## 3.1 Sources transmitting

In the first part, I will describe the first part of the communication cycle where the randomly generated binary data streams Da and Db of length Nd are encoded, modulated onto the RRC pulse, and then passed through the parametric communication channel aimed at the relay node.

### 3.1.1 Encoding

We have random binary data streams Da and Db and the sources add boundary conditions onto the streams. The conditions are that the first two bits and the last bit are all zeros. The boundary conditions are for easier decoding with the Viterbi algorithm.

Then the function *code* 3.1 gets called to get the binary coded streams Ca and Cb. The said function simply takes the data stream and using a loop it multiplies the data by function matrix G and applies mod 2 function.

```
function c = code(d)

    G = [1 1; 1 0];
    c = zeros(1, 2*length(d) - 2);

    for i=2:length(d)

        c(2*(i-1) - 1) = mod(d(i)*G(1, 1) + d(i-1)*G(1, 2), 2);
        c(2*(i-1)) = mod(d(i)*G(2, 1) + d(i-1)*G(2, 2), 2);

    end
end
```

**Figure 3.1:** Implementation of the function *code*

### 3.1.2 Modulation

Firstly we will need to create an RRC modulation pulse using a function *generate_RRC_pulse*. The function takes all the parameters needed for defining an RRC pulse and creates the pulse. Then the pulse is scaled, so his energy is equal to 1.

Secondly, we need to apply the channel function to the coded sequence and add a pilot signal at the beginning of the sequence. The CAZAC pilot signal is created using two functions *generate_cazac* and *generate_cazac_cyclic_prefix* where the second function simply creates a cyclic prefix onto already created sequence.

The channel function is very simple in BPSK modulation, that is Q = 2C - 1, for QPSK modulation the function is bit more complex and is done using function *channel_symbols*.

Lastly the created channel symbol streams `Qa` and `Qb` are sent to the function *conv_with_RRC* where the symbols get upsampled with $N_s - 1$ zeros and then are convoluted with RRC pulse. The results are the continous-time signals `Sa` and `Sb` the sources transmit in ether to the relay node.

### 3.1.3 Channel model

We have both continuous-time signals, and we apply the effect of the parametric channel to the said signals. Firstly the script generates a random phase shift `phi` with uniform distribution and time delay `tau` equal to integers of the symbol period. Then the function *channel* gets called and applies the effect of the parameters to the signal.

The function returns phase shifted and noisy signals `xA`, `xB` where first `tau` samples are pure noise. The complex noise is zero mean with variance `sigma_squared_w` equal to the value in equations 2.5 is generated using Matlab function *randn*.

## 3.2 Processing at the relay node

The relay node receives mesh of the signals `xA` and `xB` that together form a signal `x` that the relay node will work with. In this section, I will describe the received signal's processing. The following section will discuss how and what relay node broadcasts.

### 3.2.1 Pilot signal detection

Firstly the relay node will estimate which signals it received using the Neyman-Pearson theorem. I have described how it will look for our case in subsection 2.5.1. The implementation described in that subsection is in function *Neyman_Pearson* 3.2. The function takes the received signal `x`, CAZAC sequence parameters `M, N`, the false alarm probability `alpha` and SNR.

The function computes the value of threshold $\beta$, computes all values of $y$, and compares them. If the maximum value of the $y$ is larger than the threshold, the function returns 1. Otherwise, 0 is returned.

### 3.2.2 Parameters estimation

If, using the Neyman-Pearson theorem, the relay node assumes it received a useful signal, the node will try to find out the values of channel parameters using the Joint ML estimator as described in subsection 2.5.2. The implementation of the estimator is in function *estimation* 3.3.

The function firstly finds the maximum absolute value of the energy between modulated CAZAC sequence and the noisy signal `x`. The estimator estimates the time delay `tau` as a difference in samples between the peak of energy between modulated CAZAC and modulated CAZAC with prefix and the peak of computed energy between modulated CAZAC and noisy signal.

```
function value = Neyman_Pearson(x, M, N, alpha, SNR, g, N_s)

    %creating CAZAC pilot to compare x with
    cazac = generate_cazac(M, N);
    s_cazac = conv_with_RRC(cazac, g, N_s);

    %threshold computation
    sigma_squared_w = 10^(-SNR/10);
    epsilon_s = max(abs(conv(s_cazac, conj(s_cazac))));
    beta = sigma_squared_w*epsilon_s*log(1/alpha);

    y = abs(conv(x, conj(s_cazac)));

    if(max(y(:)) > beta) value = 1;
    else value = 0;
    end
end
```

**Figure 3.2:** Implementation of the function *Neyman_Pearson*

After the estimator gave us time delay estimation, the estimator gives us an angle estimation `phi` as an angle of the cross-energy in estimated `tau`.

The function returns a $2 \times 1$ array, where the first value is the estimated delay and the second is the phase.

### 3.2.3 Case for single source transmitting

If only one signal is transmitted, the relay will first use zero-forcing to remove the phase shift. The received signal is simply multiplied with the inverse estimated phase shift `x = exp(-1i*phi)*x`. After the zero-forcing of the phase shift the function *estimate_data_at_sources* is called with the inputs of noisy signal `x`, delay, parameters of the data payload: length of CAZAC `N`, `header_length` (is zero for single-source case) and length of message `Nd`. Lastly the function needs RRC pulse `g` and samples per period `N_s` to use in matched filter.

The function uses matched filter and takes its samples in multiples of the symbol period. Then based on decision regions, the function assigns a binary value to each constellation point. In BPSK modulation, the only decision criterion is the real part of the constellation point. In QPSK modulation, an imaginary part also plays a role.

### 3.2.4 Case for both signals transmitting

If both signals are assumed to be present, the relay node finds the values of channel parameters for both signals using the already described function *parameters*. With this knowledge, the relay node knows which source transmitted earlier and what is the relative time delay between both sources. Then

```
function parametres = estimation(x, M, N, g, N_s)

    cazac = generate_cazac(M, N);
    cazac_prefix = generate_cazac_cyclic_prefix(cazac);
    s = conv_with_RRC(cazac, g, N_s);
    s_prefix = conv_with_RRC(cazac_prefix, g, N_s);    %the ...
        preambule convoluted with RRC

    energy = conv(conj(s), s_prefix);
    energy_peak = find(energy(:) == max(energy(:)));

    noisy_energy = conv(conj(s), x);
    noisy_energy_peak = find(noisy_energy(:) == ...
        max(noisy_energy(:)));

    tau = N_s*round(abs(noisy_energy_peak - energy_peak)/N_s);

    energy_at_estimated_tau = 0;

    for i=1:length(s_prefix)

        if(i < length(x)-tau+1)
        energy_at_estimated_tau = energy_at_estimated_tau + ...
            x(tau+i).*conj(s_prefix(i));
        end

    end

    phi = mod(angle(energy_at_estimated_tau), 2*pi);

    parameters = [tau, phi];
end
```

**Figure 3.3:** Implementation of the function *parameters*

the relay node will apply zero-forcing to the signal.

The first zero-forcing applied is removing the CAZAC pilot signal from the later source as it is no longer needed and is only a hindrance in data detection. The function used for the removal is *remove_preamble*. the function takes the noisy signal `x` and all the necessary values for creating the delayed, phase-shifted CAZAC pilot. The parameters are: time delay `tau`, phase shift `phi`, parameters `M, N`, modulation pulse `g` and samples per period `N_s`. After the removal, the zero-forcing of the phase shift from the earlier source happens.

After the zero-forcing has been completed, we can now detect the data similarly as in the case where a single signal was transmitted. However, in this case, we need additional information regarding the relative delay and relative fading `h`. The function that detects the data in this case is *estimate_data_from_both_sources* 3.4. Inputs are noisy signals, time delays from both sources, relative fading, CAZAC, message lengths, and functions used in the matched filter.

Firstly we take the first messages of the non-overlapped data from an earlier source and, using decision regions, we transform them into binary data. The same thing is done with the non-overlapped data from the later source, except in this case, the decision regions are shifted due to the effect of relative fading `h`.

The remaining constellation points are compared with the H-constellation points generated for a given value of `h` and a detection is made once more. The function returns a binary stream of detected data.

## 3.3 Relay node broadcasting

After the data has been estimated using the matched filter, the node will broadcast the detected data. Depending on if only a single signal was transmitted or both were, the relay node might edit the detected data that will be broadcasted.

### 3.3.1 Case for single source transmitting

If only one source transmitted message, the CAZAC pilot of the transmitting signal gets added at the start of the detected sequence that turns into channel symbols and gets modulated the same way we modulated the signals from each source.

Then the continuous-time signal passes into function *channel* where randomly generated nuisance parameters get added to the useful signal.

### 3.3.2 Case for both sources transmitting

In this case, the relay node will have to first apply the xor function on the detected binary streams from the non-overlapped signal, as shown in figure 2.6.

At the start of the xorred sequence `data_relay` the CAZAC sequence of the relay node R gets added along with the header. The first half of the header with the length of `header_length` is composed of purely 1s or 0s if A or B transmitted earlier, respectively. The second half is the estimated relative delay `delta_tau` turned into binary array using Matlab function *de2bi*.

After this sequence has been completed, all bits excluding the CAZAC sequence are turned into channel symbols, modulated, and passed through the channel with functions *conv_with_RRC* and *channel*. The continuous-time signal from the relay node is called `xR`.

## 3.4 Processing at the sources

The signal `xR` passed the communication channel, and now it will be processed in the sources. Firstly the sources will use the Neyman-Pearson theorem in the form of an already defined function *Neyman_Pearson* to detect the CAZAC

```
function data_estimate = estimate_data_from_both_sources(x, ...
    tau_earlier, tau_later, h, g, N, N_s, Nd)

    MF = conv(x, g);
    data_estimate = zeros(1, Nd + (tau_later-tau_earlier)/N_s);

    %non-overlapped data from the earlier source
    MF_samples = MF(length(g) + tau_earlier + 2*N*N_s +  ...
        N_s*(0:(tau_later - tau_earlier)/(N_s)-1));
    data_estimate(1:length(MF_samples)) = ...
        0.5*sign(real(MF_samples(:))) + 0.5;

    %overlapped data (constellation points are 1+h, -1-h, ...
        1-h, -1+h)
    MF_samples = MF(length(g) + tau_earlier + 2*N*N_s +  ...
        N_s*((tau_later - tau_earlier)/(N_s):Nd - 1));

    for i=1:length(MF_samples)  %finding the closest ...
        constellation point
        if(abs(MF_samples(i) - (1-h)) < abs(MF_samples(i) - ...
            (1+h)) &&...
                abs(MF_samples(i) - (1-h)) < ...
                    abs(MF_samples(i) - (-1-h)) ||...
                abs(MF_samples(i) - (-1+h)) < ...
                    abs(MF_samples(i) - (1+h)) &&...
                abs(MF_samples(i) - (-1+h)) < ...
                    abs(MF_samples(i) - (-1-h)))

                data_estimate((tau_later-tau_earlier)/N_s + ...
                    i) = 1;
         end
    end

    %non-overlapped data from the later source
    MF_samples = MF(length(g) + 2*N*N_s + tau_earlier +  ...
        N_s*(Nd: Nd + (tau_later-tau_earlier)/N_s-1));

    for i=1:length(MF_samples)

    if(abs(MF_samples(i) - h) < abs(MF_samples(i)+h))
                data_estimate(Nd+i) = 1;
    end
    end
end
```

**Figure 3.4:** Finding closest H-Constellation point in the function *estimate_data_from_both_sources* for BPSK modulation

pilots. After the signal detection has happened, the channel parameters are estimated using function *estimation*. Then two types of data estimating will happen - a hard decision detection using decision regions in constellation space and a soft decision decoding using constellation space distance as a metric.

### 3.4.1 Hard decision data detection

In this part, the sources will use decision regions in constellation space. The sources will act as if the transmitted sequences Ca and Cb were uncoded messages, so the error of the uncoded communication can be easily computed.

#### A single signal case

If the sources assumed the relay node acted as a simple forwarder, the sources would apply zero-forcing of the phase shift estimated in the previous step. After that the function *estimate_data_at_sources* gets called. The last input of the function has the value '0'. This value is the header's length, an additional number of bits the function needs to return. In the single signal case, the header is not present in the signal, so the value is 0. After the detection the coded array Ca or Cb is compared with the true transmitted coded sequences Ca or Cb to give us an error rate of the "uncoded" communication.

#### Multiple signals case

If multiple signals were transmitted as is signified by the CAZAC pilot typical for the relay node, the start of the procedure remains the same. The zero-forcing of the phase shift happens, and the same function *estimate_data_at_sources* gets called except now the value of the header length is the defined non-zero value header_length the reader can modify in the script.

The function returned us an array data_sources of length Nc + header_length and now the sources will extract the information from the header. After the sources know which one transmitted first and their relative delay from the header, the comparison with the true transmitted sequences Ca and Cb happen. In this case, the circular shift needs to be applied to the transmitted sequences. The resulting errors of the comparison are called error_A and error_B and are displayed on the console.

### 3.4.2 Decoding

The sources will be using the Viterbi algorithm for decoding the acquired signal with the metric of the algorithm being the distance in the constellation space. Then the decoded data will be compared with the actual transmitted data Da and Db to give us an error performance.

#### Single signal case

If only a single signal was transmitted, the decoding happens purely with the use of a function *Viterbi*. This function takes the matched filter outputs and the length of the message. The output is the decoded message Da_est or Db_est.

In the function, the forward phase happens in the first for loop where matrix trellis is filled. The value trellis$(i, j)$ gives us minimal value of metric to get to the $i$-th state in $j$-th step. The backward phase happens after the first loop has ended, where the array `states` is filled. This array represents which states the encoder has passed in encoding the initial sequence.

The implementation of the two phases in *Viterbi* function for BPSK modulation is shown in figure 3.5.

After the function *Viterbi* has returned the decoded data the error rates `error_A_soft_decision` or `error_B_soft_decision` are computed.

### Multiple signals case

The procedure for multiple signals starts with applying the inverse XOR function to the constellation space points representing the received signal. The is done using the function *xor_signal*. The function is a simple for loop that does the decision-making described in tables 2.1 and 2.2.

After the function returned us constellation space arrays `x_xorred_A` and `x_xorred_B`, we apply circular shift to the points of the signal that has transmitted earlier. After all this has been done, the rest of the procedure is the same as with the case for a single signal. The function *Viterbi* gets called, and the error rates are computed and displayed.

```
for i=2:Nd  %forward phase, getting rellis matrix

    %for getting into state 0
    rho1 = abs(x(2*(i-1) - 1) + 1) + abs(x(2*(i-1)) + 1);
    rho2 = abs(x(2*(i-1) - 1) - 1) + abs(x(2*(i-1)) + 1);

    trellis(1, i) = min(trellis(1, i-1) + rho1,...
        trellis(2, i-1) + rho2);


    %for getting into state 1
     rho1 = abs(x(2*(i-1) - 1) - 1) + abs(x(2*(i-1)) - 1);
     rho2 = abs(x(2*(i-1) - 1) + 1) + abs(x(2*(i-1)) - 1);

     trellis(2, i) = min(trellis(2, i-1) + rho2,...
         trellis(1, i-1) + rho1);


end
```

**(a) :** Forward phase of Viterbi algorithm

```
for i=Nd:-1:2

    if states(i) == 0

       rho1 = abs(trellis(1, i) - trellis(1, i-1) - ...
           abs(x(2*(i-1) - 1) + 1) - abs(x(2*(i-1)) + 1));
       rho2 = abs(trellis(1, i) - trellis(2, i-1) - ...
           abs(x(2*(i-1) - 1) - 1) - abs(x(2*(i-1)) + 1));

       if rho1 < rho2
        states(i-1) = 0;
       else states(i-1) = 1;
       end

    else

        rho1 = abs(trellis(2, i) - trellis(2, i-1) - ...
            abs(x(2*(i-1) - 1) + 1) - abs(x(2*(i-1)) - 1));
        rho2 = abs(trellis(2, i) - trellis(1, i-1) - ...
            abs(x(2*(i-1) - 1) - 1) - abs(x(2*(i-1)) - 1));

        if rho1 < rho2

            states(i-1) = 1;
        else states(i-1) = 0;
        end
    end
end
```

**(b) :** Backward phase of Viterbi algorithm

**Figure 3.5:** Implementation of the function *Viterbi* for BPSK modulation

# Chapter 4

# Simulation and results

This chapter shows the numerical results of simulation done in attached files. First sections focus on detection and parameter estimation, their accuracy and how close they approach the theoretical best performance. The later sections focus on error performance for the case of both sources transmitting. The error performance is computed as a comparison between transmitted data from the sources and the estimated data at the sources.

All presented graphs have been plotted in Matlab and the `.m` files are included with this document.

## 4.1 Signal detection

Firstly I will talk about a rate at which the Neyman-Pearson detector accept the alternative hypothesis. The detector accepts the hypothesis if $|\langle \boldsymbol{x}, \boldsymbol{s}\rangle|^2 > \sigma_w^2 \epsilon_s^2 \ln\left(\frac{1}{\alpha}\right)$ as computed in subsection 2.5.1.

I have generated a CAZAC sequence of length $N = 37$, and in each step, random noise was added to the signal. After that, the function *Neyman_Pearson* 3.2 was called to either accept or reject the alternative hypothesis with various values of false alarm probability $\alpha$. The rate at which the detector accepted the alternative hypothesis is plotted below in figure 4.1.

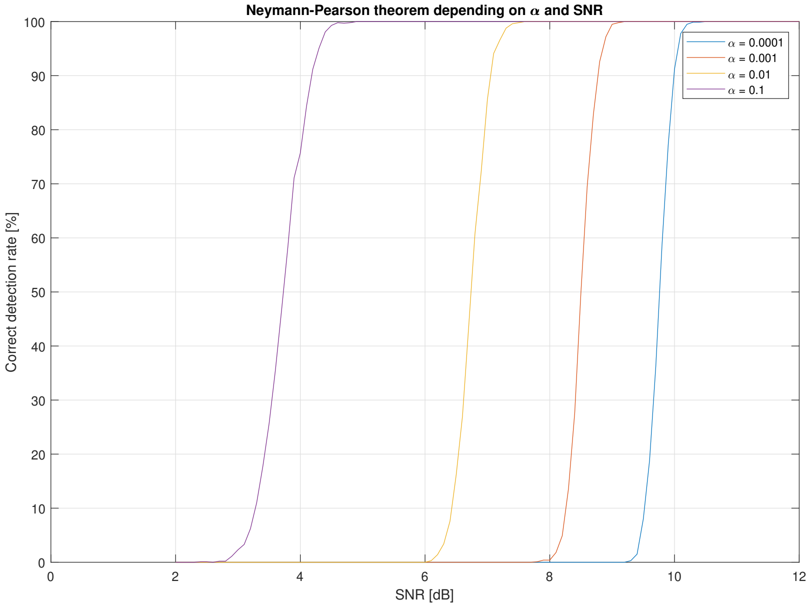


**Figure 4.1:** Correct detection dependence on SNR and $\alpha$

As we can see, the detection for each value of $\alpha$ has a certain SNR threshold where the detection accuracy sharply drops, and beyond that point, the detection rate is 0%. The smaller the value of $\alpha$, the smaller the SNR threshold is. The SNR threshold corresponds with the $\sigma_w^2 \ln\left(\frac{1}{\alpha}\right) \approx 1$.

The reader can find this script as **`Neyman_Pearson.m`** in attached files.

## 4.2 Channel parameter estimation

This section will talk about how the accuracy of the estimation fares to the theoretical peak and how much accuracy decays with increasing noise. Firstly I will show the accuracy of the delay estimation and how it can be affected by a non-orthogonal signal in the form of modulated binary data. Then we will take a look at how the phase estimators' variance fares compared to the Cramer-Rao Lower Bound.

### 4.2.1 Delay estimation

The first parameter we will look at is the estimation of the random delay equal to the integers of symbol period $N_s$. The Jointed ML estimator finds the peak of cross-energy between the noisy signal that has a modulated CAZAC sequence in it and the sequence itself as described in 2.13.

For each value of SNR, the estimation has been made $10^4$ times. In figure 4.2 we can see the error rates of delay estimation for 4 different signals. The first two signals use a shorter CAZAC sequence containing only $N = 37$ symbols, and the second two use sequence of length $N = 73$ (excluding the prefix).

Each CAZAC sequence has been sent as an individual signal, and part of a payload consisting of the modulated CAZAC itself and BPSK modulated binary data. The PSK modulation is not orthogonal with CAZAC, so the data can affect the delay estimation accuracy.

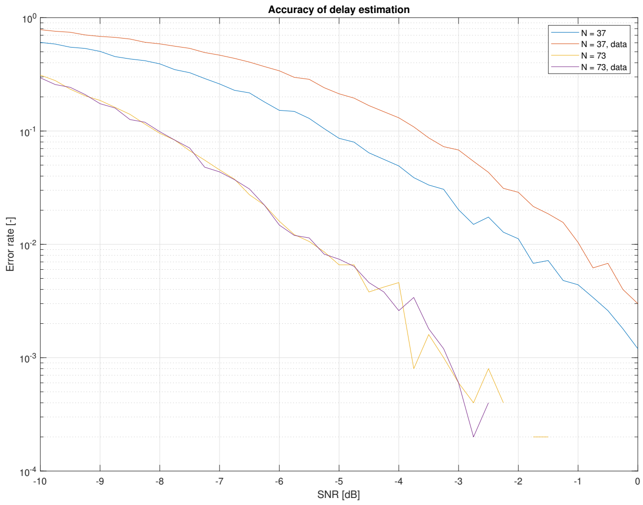


**Figure 4.2:** Accuracy of delay estimation using Joint ML estimator

We can see in that the accuracy of the shorter sequence is worse compared to the longer one. Also, the shorter sequence shows worse results if the binary data were also transmitted with it, whereas the longer sequence shows basically no decline in performance with data added.

The figure has been plotted using script `delay_est_ML_detector.m`.

### 4.2.2 Phase estimation

The following parameter we need to estimate is the phase shift equal to the angle of the correlation function at the estimated delay as shown in 2.14. The computation is repeated in many steps, and the resulting histogram of the estimation will be compared with the theoretical best distribution function - Gaussian curve with the mean equal to the true value of the phase shift and variance equal to the Cramer-Rao Lower Bound.

If the delay estimation is incorrect by many symbol periods, the phase shift estimation will be a random number that has no relation to the phase shift. So, for any computation of the following graphs, a perfect knowledge of the delay is assumed.

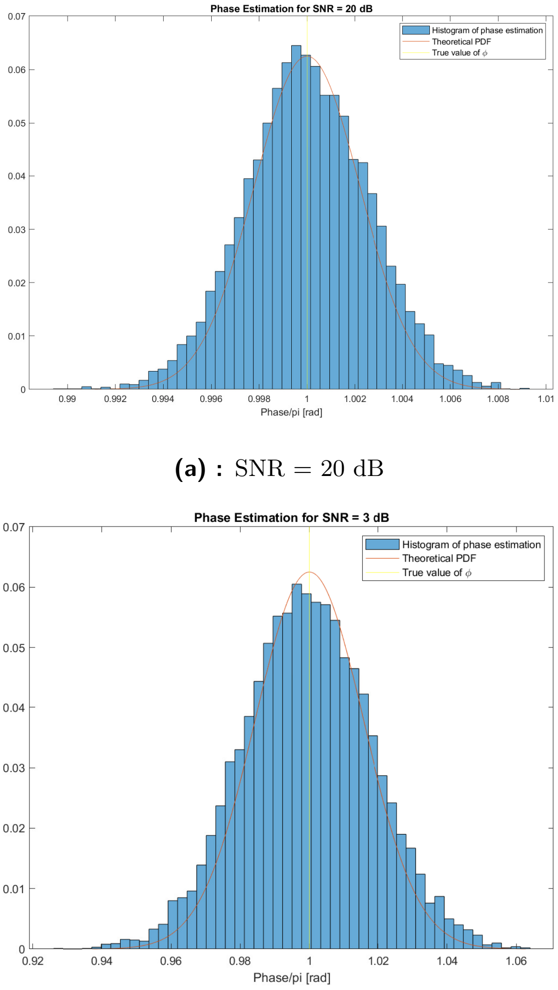
#### Single pilot case

We will start with the case of a single pilot in the ether that is only affected by AWGN. A Cramer-Rao Lower Bound gives the theoretical best variance the estimation can provide. The lower bound computed in 2.5.2 was 2.20. Using this equation for two values of SNR, we get the following values of estimation deviation shown in table 4.1.

| SNR | Standard deviation |
|---|---|
| 20 dB | $0.0022\pi$ |
| 3 dB | $0.0157\pi$ |

**Table 4.1:** Relation of the SNR to the standard deviation for single pilot case

We can compare the theoretical deviation with the estimation histogram in figure 4.3. The histograms match the Gaussian curve fairly accurately, both with mean and variance. For more computations, the histograms would approach the curve even more tightly.



**(a) :** SNR = 20 dB

**(b) :** SNR = 3 dB

**Figure 4.3:** Phase estimation histograms compared with ideal distribution function for single CAZAC pilot

### Multiple pilot case

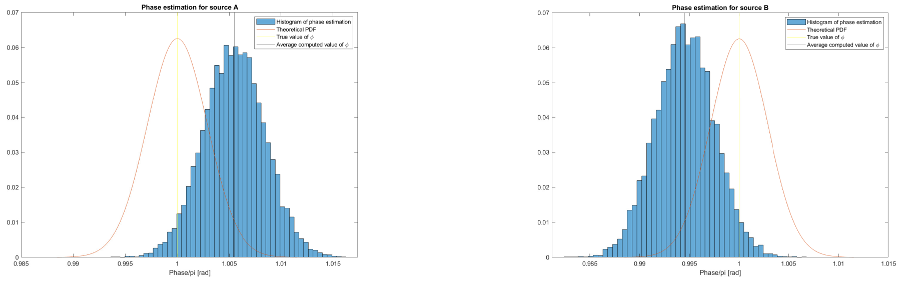
For multiple overlapping pilots the CRLB gave us 2.22 where $\eta$ is a scaled cross energy defined in 2.23. The worst-case for variance can happen if the true values of $\varphi_A, \varphi_B$ happen to be the same. For my case, the scaled cross energy was $\eta = 0.165$ and true phase shift values were equal. The computations were done for case with SNR = 3 dB and SNR = 20 dB. The table 4.2 shows the standard deviation the CRLB gives us.

| SNR | Standard deviation |
|---|---|
| 20 dB | $0.0029\pi$ |
| 3 dB | $0.0204\pi$ |

**Table 4.2:** Relation of the SNR to the standard deviation for multiple pilot case

Comparing the standard deviation values to the same values for a single pilot case shows about a 30% rise in value, thus degrading the estimator.

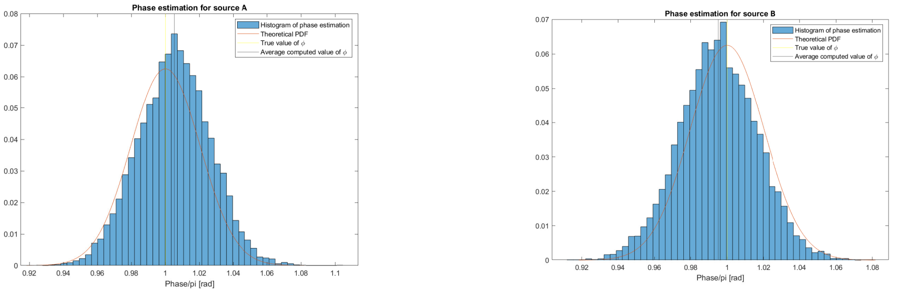
Another degradation the multiple non-orthogonal pilots cause is the shift between the true value of phase shift and the average estimated value of the said parameter. This shift becomes very apparent for the case with small deviation and thus very steep Gaussian PDF for the case with SNR = 20 dB in figures 4.4. In the case with SNR = 3 dB in figures 4.5 the shift is less apparent.



**(a) :** For source A

**(b) :** For source B

**Figure 4.4:** Phase estimation histograms for SNR = 20 dB



**(a) :** For source A

**(b) :** For source B

**Figure 4.5:** Phase estimation histograms for SNR = 3 dB

The scripts used for plotting histograms are `CRLB_single_pilots.m` and `CRLB_two_pilots.m`.

## 4.3 Error performance

In this section, I will show the resulting *error performance* of the communication topology in relation to the SNR and the angle of the relative fading $h$. In the following graphs, a perfect knowledge of the channel parameters was assumed as the possible imperfection of the estimation can affect the error performance.

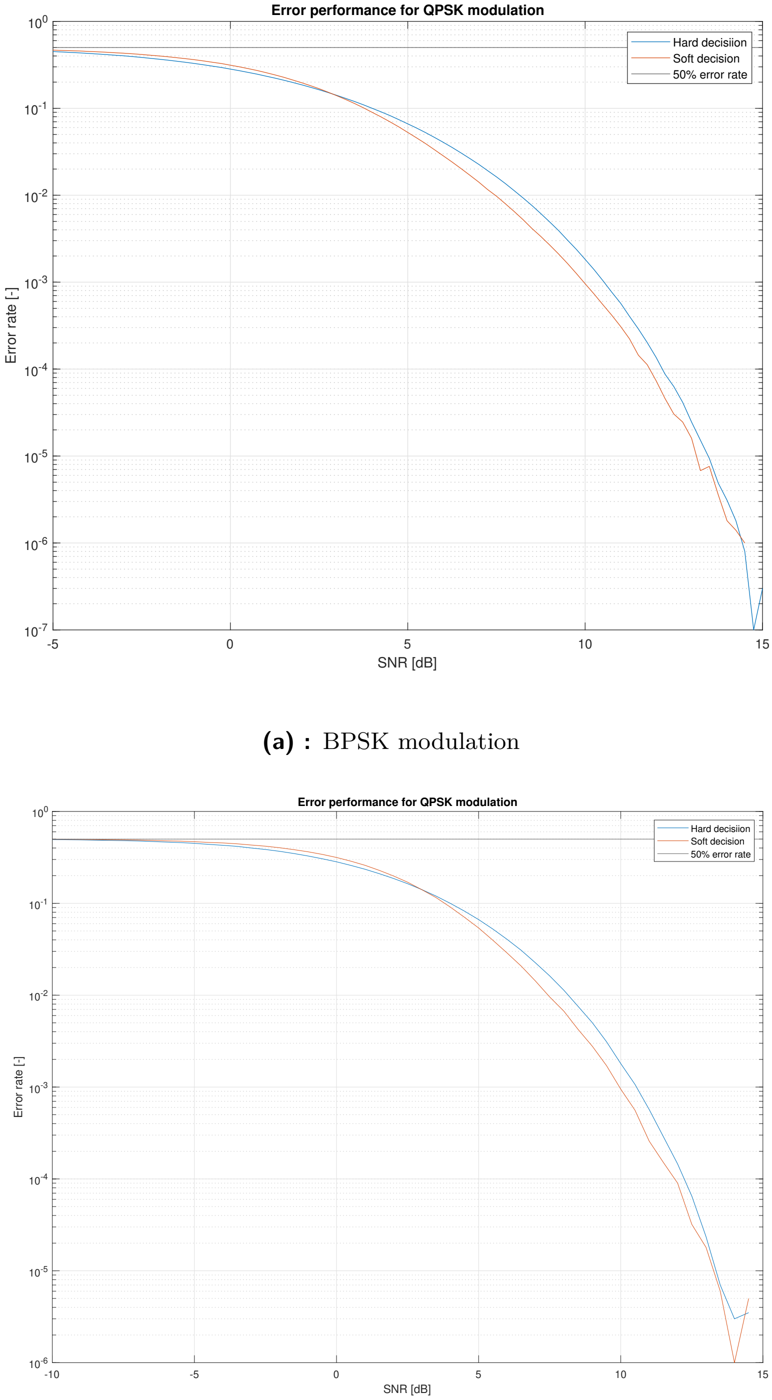
The error bits in all computations are wrongly detected/decoded bits at the sources in the case of both sources transmitting. The bits could have been wrongly detected at the relay node or at the sources throughout the communication cycle. The simulations only compared the messages the sources truly transmitted and the message the other source detected/decoded.

### 4.3.1 SNR dependence

We most commonly look at the signal error performance dependence on the Signal-to-Noise Ratio related to the AWGN noise variance $\sigma_w^2$. The error rates have been computed using for loop repeatedly detecting the data, then storing the error rates, and finally taking the mean error rate. In every step of the loop, the useful signals have been affected by complex AWGN with variance computed in equations 2.5.

The two graphs shown in figure 4.6 give us the error rates for both BPSK and QPSK modulation. We can see that for very low SNRs around -5dB, the error performance approaches the absolute worst performance threshold - a 50% error rate. Communication is out of the question, and the detected/decoded data is essentially randomly generated with no relation to the transmitted data. We can also notice that the error performance of a coded message is slightly better than that of an uncoded one.

The graphs have been plotted using scripts `SER_SNR_BPSK.m` and `SER_SNR_QPSK.m`.



**(a) :** BPSK modulation

**(b) :** QPSK modulation

**Figure 4.6:** Error performance dependence on SNR

### 4.3.2 Relative fading dependence

As described in 1.9.3, the relative fading $h$ has a significant effect on the H-constellation of the relay node and thus affects the *free distance* of the constellation. The smaller the free distance, the weaker noise needed for the information to be correctly estimated. This dependence will be shown in the following graphs for QPSK modulation along with H-constellations for angles of $h$ close to the "problematic" angles.

As we can see from the figure 4.7, the performance quickly decays for angles close to $\{\pm\frac{\pi}{2}\}$. This decay is especially obvious for very high SNRs. The reason is the overlapping constellation points in H-Constellation that produce different XOR products. With free distance approaching zero, even very slight noise can cause an error in processing.

The H-constellation in figure 4.8b shows this very well. For $h$ approaching the j the constellation points representing different XOR function outputs will overlap and with that the relay node cannot distinguish what was transmitted. The H-constellation for $h$ close to 1 in figure 4.8a shows the opposite. The constellation points representing same outputs are close to each other.

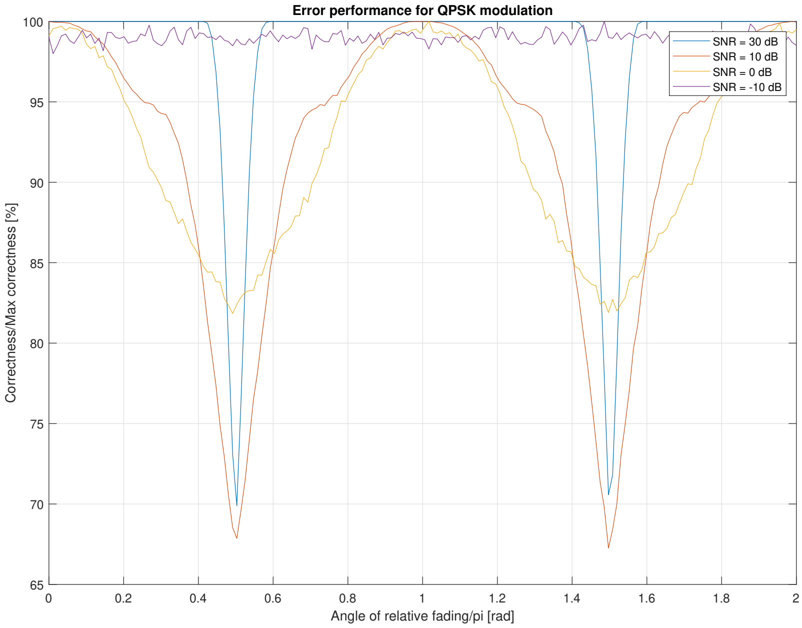
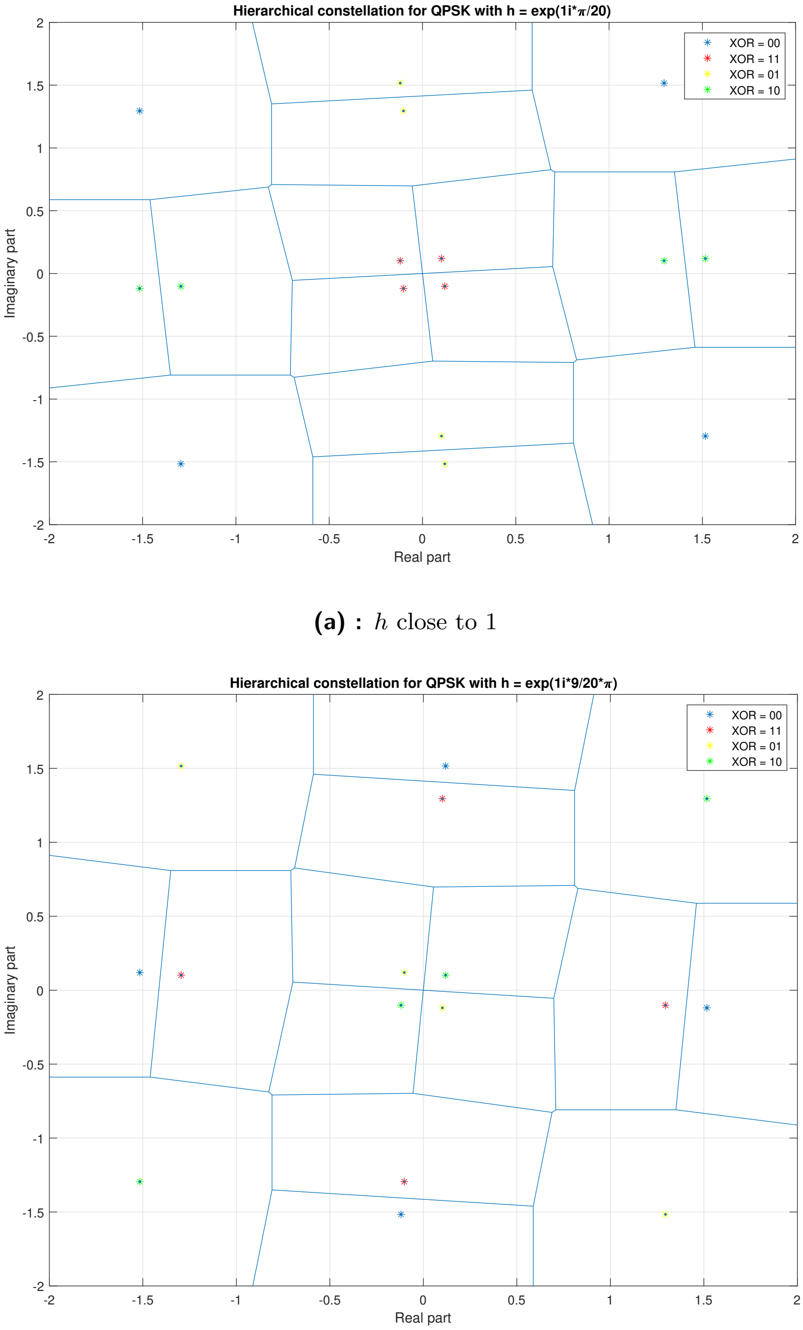


**Figure 4.7:** Error performance dependence on angle of relative fading $h$ for QPSK modulation

The computation of error performance can be found in script `SER_dependent_on_h_QPSK.m`. The Voronoi diagram representing the H-constellations can be found in `Voronoi_QPSK.m`.



**(a) :** $h$ close to 1

**(b) :** $h$ close to j

**Figure 4.8:** H-Constellations for certain values of $h$ for QPSK modulation

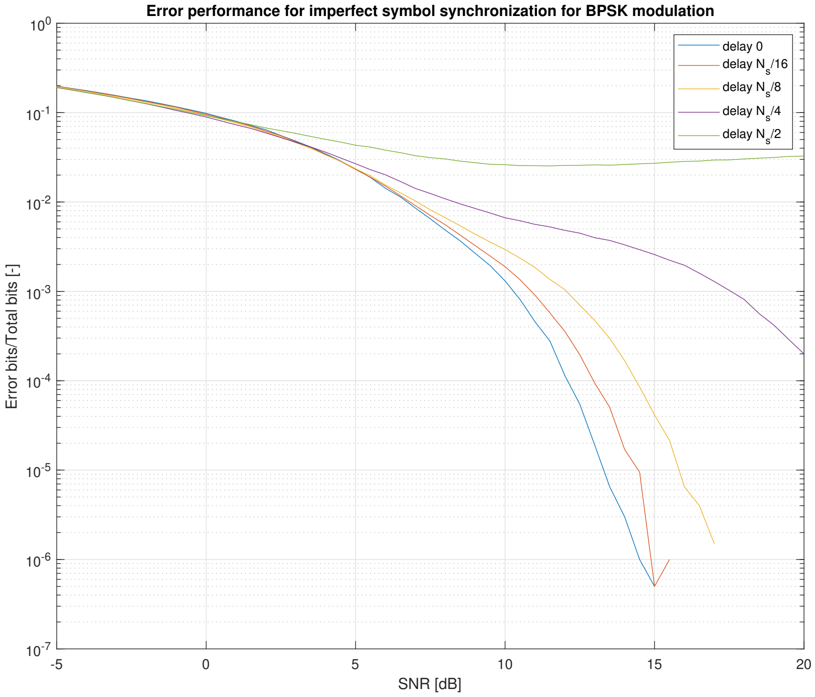
## 4.4 Imperfect symbol period synchronization

The last performance criterion I will describe is the effect of the imperfect symbol period synchronization. A sample perfect symbol synchronization was assumed for the past computations and in the main scripts. This section will show the effects caused if this condition is not satisfied.

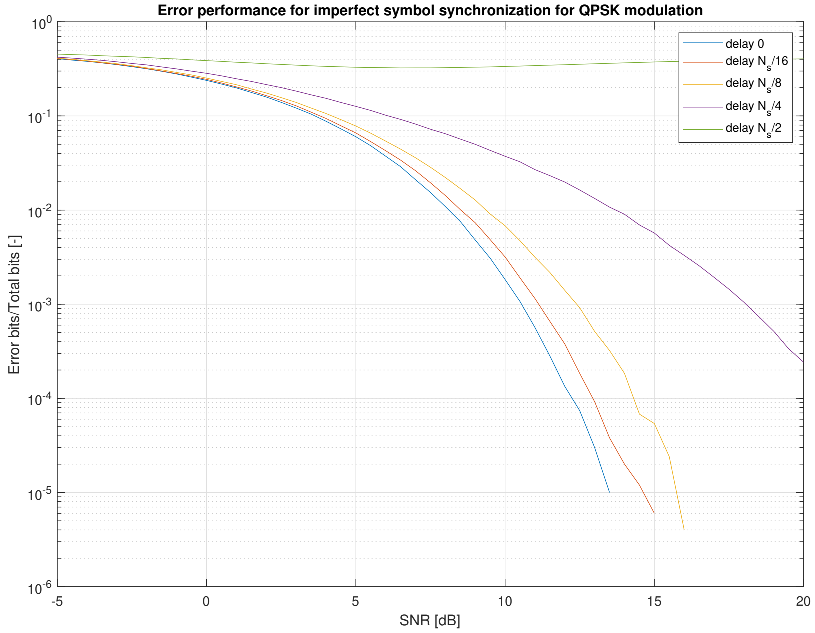
The following computations were done with the samples per period $N_s = 16$, and the error performance is dependent on SNR and the relative sample shift of symbols from both sources. Relative fading $h$ was equal to 1.

The results in figure 4.9 show that the performance decays with an increasing relative delay between symbols. The worst case is for the relative delay to be equal to half of the $N_s$ where the error performance does not improve with higher SNR. The cause is that some combinations of data transmitting cause the matched filter output to be very close to the border of the decision regions.

The computations are done in scripts `imperfect_synchronization_BPSK.m` and `imperfect_synchronization_QPSK.m`.



**(a) :** BSPK modulation



**(b) :** QPSK modulation

**Figure 4.9:** Effect of the imperfect synchronization on error performance

# Chapter 5
# Conclusion

This thesis aimed to get acquainted with the fundamentals of WPNC and estimation and detection theory and apply them in a simple network topology.

In the first chapter, the reader was introduced to the necessary theoretical background. The second chapter focused on applying the previous fundamentals in a topology "Two-Way Relay Channel" with BPSK or QPSK modulation. This chapter provided the analytical expression for the phase and delay estimator that used the known CAZAC sequence as a pilot signal. The third chapter focused on implementing said network topology in the Matlab program. The algorithms were described in detail, along with occasional examples of the code. The last chapter demonstrates the numerical results showing the signal detector and estimator performance. Next, the overall error performance for the case of both signals transmitting is shown, for which the perfect channel parameters knowledge was assumed.

The simulations showed that the ML phase estimator is asymptotically optimal for large records. The estimation histograms were compared with a Gaussian curve with a mean equal to the true value of the parameter and variance equal to the CRLB. The histograms matched the curve, as the theory says. Next, it was shown how the coding improves the error performance, even when the used coding is very simple. Also, the simulation showed how the performance decays when the angle of the relative fading is close to specific angles. The explanation of these performance drops was given.

My contribution is the Matlab implementation of the Neyman-Pearson signal detector and ML phase and delay estimator using the known modulated CAZAC sequence together with convolutional encoder and decoder using the *Viterbi algorithm.*

Further work could focus on more complicated channel models or network topologies with three or more sources. Also, a focus can be on adaptive detecting on the relay node with QPSK modulation.

# Bibliography

[1] PROAKIS, John G. *Digital communications.* 5th ed. Boston: McGraw-Hill, 2008. ISBN 978-0-07-295716-7

[2] SYKORA, Jan and Alister BURR. *Wireless Physical Layer Network Coding.* Cambridge University Press, 2018. ISBN 978-1-107-09611-0

[3] KAY, Steven M. *Fundamentals of statistical signal processing, Volume 1 Estimation Theory.* Upper Saddle River: Prentice Hall, 1993. ISBN 0-13-345711-7

[4] KAY, Steven M. *Fundamentals of statistical signal processing, Volume 2 Detection Theory.* Upper Saddle River: Prentice Hall, 1998. ISBN 978-0135041352.

[5] SYKORA, Jan. *Digital Communications - lecture slides.* 2016

[6] SYKORA, Jan. *Statistical Signal Processing - lecture slides.* 2017

# Appendix A

# Attached Matlab files

```
main scripts
│ ├── TWRC_BPSK_coded.m
│ └── TWRC_QPSK_coded.m
└── scripts for plotting graphs
    ├── CRLB_single_pilot.m
    ├── CRLB_two_pilot.m
    ├── delay_est_ML_detector.m
    ├── imperfect_synchronization_BPSK.m
    ├── imperfect_synchronization_QPSK.m
    ├── Neyman_Pearson.m
    ├── SER_h_BPSK.m
    ├── SER_h_QPSK.m
    ├── SER_SNR_BPSK.m
    ├── SER_SNR_QPSK.m
    └── Voronoi_QPSK.m
```