Bachelor Project

Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Radioelectronics

WPNC Coding and Processing in Simple Wireless Networks

Michal Šimek
Open Electronic Systems

Supervisor: Prof. Ing. Jan Sýkora, CSc.
May 2018

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Šimek Michal** Personal ID number: **457249**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Radioelectronics**

Study program: **Open Electronic Systems**

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

WPNC Coding and Processing in Simple Wireless Networks

Bachelor’s thesis title in Czech:

WPNC kódování a zpracování v jednoduchých bezdratových sítích

Guidelines:

Student will get acquainted with fundamentals of Wireless Physical Layer Network Coding (WPNC) in simple scenarios including two-way relay channel and butterfly network and with focus on hierarchical decode and forward strategy. The work should include theoretical performance analysis, practical design and computer simulation verification of the Network Coded Modulation (NCM) and related coding/decoding/processing for butterfly network. NCM should use a coded layered isomorphic design with some simple (e.g. convolutional) outer code, a variety of inner constellation alphabets/mappings, and simplified scenario model (perfect symbol timing synchronous network). A special attention should be paid to relative channel parameterisation impact on the end-to-end error rate performance. Depending on the availability of experimental TxR, selected parts of the processing should also be practically verified in over-the-air experiment.

Bibliography / sources:

[1] J. Sykora, A. Burr: Wireless Physical Layer Network Coding, Cambridge University Press 2018

Name and workplace of bachelor’s thesis supervisor:

prof. Ing. Jan Sýkora, CSc., Department of Radioelectronics, FEE

Name and workplace of second bachelor’s thesis supervisor or consultant:

Date of bachelor’s thesis assignment: **31.01.2018** Deadline for bachelor thesis submission: **25.05.2018**

Assignment valid until: **30.09.2019**

prof. Ing. Jan Sýkora, CSc.
Supervisor’s signature

Head of department’s signature

prof. Ing. Pavel Ripka, CSc.
Dean’s signature

III. Assignment receipt

The student acknowledges that the bachelor’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor’s thesis, the author must state the names of consultants and include a list of references.

.

Date of assignment receipt

Student’s signature

Acknowledgements

I would like to thank my supervisor prof. Ing. Jan Sýkora, CSc., for helping me and giving me the opportunity to explore the basics of WPNC coding. I would also like to thank my family and all others for supporting me during 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.

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 thesis deals with Wireless Physical Layer Network coding. The aim is to get acquainted with this issue, which takes into account the behavior of electromagnetic waves in the same physical space and works with it. By using this approach, we can achieve better system throughput and performance. Next part of the study is the implementation and simulation of error rate in Two-way relay channel, which is a special case of so-called "butterfly" network. Topology with three source nodes that use BPSK modulation, and one relay node with focus on the HNC map and the H-constellation at the relay node is examined. Briefly, a few techniques are summarized on how to determine the quality of the HNC map for a given channel parameterization.

Keywords: WPNC coding, Butterfly network, three-source-node network topology, Network coded modulation

Supervisor: Prof. Ing. Jan Sýkora, CSc.

Abstrakt

Tato bakalářská práce se zabývá kódováním na fyzické vrstvě sítí. Cílem je seznámení se s touto problematikou, která bere v potaz chování elektromagnetických vln ve stejném fyzikálním prostoru a pracuje s tím. Výsledkem je lepší propustsnost a chování systému. Součástí práce je implementace a simulace chybovosti v topologii Two-way relay channel, která je speciálním případem tak zvané "motýlí" topologie. Dále je zkoumána topologie se třemi zdrojovými uzly, které používají BPSK modulaci, a jedním přeposílacím uzlem se zaměřením na HNC mapu a H-konstelaci na tomto uzlu. Krátce jsou nastíněny nápady, jak určit kvalitu HNC mapy pro danou parametrizace kanálu.

Klíčová slova: WPNC kódování, motýlí síť, topologie se třemi zdrojovými uzly, síťově kódovaná modulace

Překlad názvu: WPNC kódování a zpracování v jednoduchých bezdrátových sítích

Contents

Figures

Tables

Chapter 1
Introduction

Wireless Physical Layer Network (WPNC) coding is intensively under the research. It is a promising way how to deal with dense radio networks. One of the biggest problems in wireless communications is interference. If multiple sources transmit signals which are not orthogonal, they are superimposed. Nowadays, this problem is mainly solved by applying multiple access techniques such as FDMA (Frequency Division Multiple Access) or TDMA (Time Division Multiple Access). WPNC coding takes into account the '*mixing*' of electromagnetic waves and works with it.

1.1 Brief history of communications

Being able to communicate is one of the greatest need in human's life. In the history of mankind, countless approaches have been invented and used to send messages to far distances starting from the simplest ones such as smoke signaling. At the beginning of 19th century in France, during Napoleon wars, the optical telegraph was on the rise and it was the best long-distance communications technique at the time. However, it had few disadvantages. It was dependent on weather and daylight [4]. The branch of communications has been developing rapidly in the 19th and 20th century. The first successful transmission of message via a transatlantic telegraph cable was sent in the August of 1858 [5]. Eighteen years later, Alexander Bell submits the world's first telephone patent, and in the same year, as the first one in the world, he calls over the telephone and says the famous sentences *"Mr. Watson. Come here. I want to see you."* [4] After developments in electromagnetism, ideas of radio transmission come to life. Italian physicist Guglielmo Marconi sent radio transmission of the Morse-code signal for the letter "s" across Atlantic ocean. Satellite transmission began in the middle of the 20th century. Since then, lots of improvements and innovations have been done in wireless communications and data transmission.

1.2 Outline

The thesis is divided into five chapters. Chapter 2 describes the theoretical minimum in digital communications that is required to be introduced in order to understand the procedures and processes in Wireless Physical layer Network Coding (WPNC). A brief introduction of convolutional codes is also mentioned. In the next chapter, we are going to summarize basic properties and definitions of WPNC coding. The fourth chapter focuses on three-source-node networks where WPNC coding can be applied. Lots of problems are unexplored in this topology. Finally, we utilize our knowledge to simulate transmission and compute error rate performance in Two-way relay channel.

Chapter 2

Fundamentals of digital communication

In this chapter, basic knowledge of digital communications will be introduced. We will define constellation space, basic properties of convolutional code and its decoding and how to deal with AWGN communication channel.

Modulators are blocks that take discrete data d_n and produce continuous function that can be broadcasted in the form of electromagnetic waves. We assume modulation

$$s(t) = \sum_n g(q_n(d_n, \sigma_n), t - nT_S), \tag{2.1}$$

where $s(t)$ is a complex envelope of modulated signal, $g(t - nT_S)$ is symbol pulse which is a function of channel symbol q_n that depends on actual data d_n and some modulator state σ_n. T_S is a symbol period.

In this thesis, we will deal with linear modulation without memory (states of modulator). Then we can define it as follows

$$s(t) = \sum_n q_n(d_n) g(t - nT_S). \tag{2.2}$$

Very important attribute for the pulses $g(t - nT_S)$ to fulfill is the Nyquist condition.

Definition (Nyquist condition). We say that the pulses $g_1(t - kT_S)$ and $g_2(t - lT_S)$ are Nyquist, if the following holds

$$\int_{-\infty}^{\infty} g_1(t - kT_S) g_2^*(t - lT_S) \mathrm{d}t = 0, \qquad \forall k \neq l \tag{2.3}$$

where $*$ denotes complex conjugate [2].
In other words, the pulses are Nyquist, if they are orthogonal everywhere except when they have the same time period. Sometimes, we require the modulation itself to be orthogonal.

Definition (Nyquist modulation). We assume liner modulation with symbol pulse $g(t)$. The modulation is called Nyquist iff

$$\int_{-\infty}^{\infty} g(t-lT_S)g^*(t-kT_S)\mathrm{d}t = 0, \qquad \forall k \neq l \tag{2.4}$$

2.1 Constellation space

Constellation space is a representation of modulated signal in orthonormal signal space [2]. We assume to have Nyquist modulation and a modulated signal $s(t)$. Let $\beta_{n,i}(t)$ be the orthonormal basis of the modulated signal

$$\{\beta_{n,i}(t)\}_{n,i} = \{\beta_i(t-nT_S)\}_{i=1}^{N_S}, \tag{2.5}$$

where N_S is the dimensionality of the modulation and T_S is symbol period. Then for every channel symbol q_n, there exists a vector in constellation space $\boldsymbol{s}_n$ such that

$$\boldsymbol{s}_{n,i} = \langle s(t), \beta_i(t-nT_S)\rangle_{L_2} = \int_{-\infty}^{\infty} s_n(t)\beta_i^*(t-nT_S)\ dt. \tag{2.6}$$

Under the assumption of Nyquist modulation, orthonormal pulse g_i can be interpreted as the basis

$$\beta_i(t-nT_S) = g_i(t-nT_S). \tag{2.7}$$

Then for points in constellation space we can write

$$s_{n,i} = q_{n,i}, \qquad i \in \{1,...,N_S\}. \tag{2.8}$$

In addition, if the modulation is linear (that is $N_S = 1$) then $s_n = q_n$.

2.1.1 PSK modulation

PSK is an abbreviation for Phase Shift Keying and it is a linear modulation. It shifts the angle of constellation points q_n.

$$q_n \in \{\mathrm{e}^{j\frac{2\pi}{M_q}i}\}_{i=0}^{M_q-1}, \tag{2.9}$$

where M_q is number of symbols. It basically deploys symbols q_n on unit circle in constellation space. In this thesis, we will use the simplest PSK modulation called Binary Phase Shift Keying (BPSK). It places two symbols on real axis to points

$$q_n \in \{-1, 1\}. \tag{2.10}$$

The second modulation which appears in our simulation is a special case of PSK called Quaternary Phase Shift Keying (QPSK). It is defined as

$$q_n \in \left\{\frac{1+j}{\sqrt{2}}, \frac{-1+j}{\sqrt{2}}, \frac{1-j}{\sqrt{2}}, \frac{-1-j}{\sqrt{2}}\right\}. \tag{2.11}$$

2.2 Convolutional codes

There are plenty of disturbing elements such as noise in a communication channel. In many applications, including digital communications, it is required to achieve reliable data transmission. This gave rise to coding theory. Typically, the code augments the data by redundant information, and after a message is sent through a noisy communication channel, it can detect and correct a limited number of errors. The convolutional code is an example of error-correcting code.
Let $\boldsymbol{b}_n$ be a word of length N_b and $\boldsymbol{c}_n$ an encoded word of length N_c, then

$$\boldsymbol{c}_n = \sum_{i=0}^{N} \boldsymbol{G}_i \boldsymbol{b}_{n-i}, \tag{2.12}$$

$$R = \frac{N_b}{N_c} \tag{2.13}$$

where $\boldsymbol{G}_i$ is a generator matrix of size N_c x N_b , N is called constraint length and R denotes code rate [2]. The code rate R reveals the ratio between the input and output sequence. It basically says the length of output word with respect to input. Let's demonstrate the principle on a simple example.

Example 1. Let $N_b = 1$ and $N_c = 2$. The code rate is then $R = \frac{1}{2}$, which means that from one input bit, new two bits are computed. We set the generator matrices to

$$\boldsymbol{G}_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \ \boldsymbol{G}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}. \tag{2.14}$$

Constraint length $N = 1$. The goal is to encode the following sequence

$$\boldsymbol{b} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}. \tag{2.15}$$

The encoded symbols are computed according to equation 2.12.

$$\boldsymbol{c}_1 = \boldsymbol{G}_0 b_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} 1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{2.16}$$

$$\boldsymbol{c}_2 = \boldsymbol{G}_0 b_2 + \boldsymbol{G}_1 b_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} 0 + \begin{pmatrix} 1 \\ 1 \end{pmatrix} 1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \tag{2.17}$$

$$\boldsymbol{c}_3 = \boldsymbol{G}_0 b_3 + \boldsymbol{G}_1 b_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} 1 + \begin{pmatrix} 1 \\ 1 \end{pmatrix} 0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{2.18}$$

The resulting encoded sequence is

$$\boldsymbol{c}^T = \begin{pmatrix} 10 & 11 & 10 \end{pmatrix} \tag{2.19}$$

2.2.1 Viterbi algorithm

Decoding convolutional code is not a simple task. The problem is that the convolutional code contains memory. The first idea is to take all states $\boldsymbol{\sigma}_{\mathrm{n}}$, find all possible combinations of data $\boldsymbol{d}_n$ and compare them to received data. It can be shown that this procedure will always work. However, its time complexity exponentially increases with message length. The most used decoding algorithm of the convolutional code is called Viterbi algorithm. It works with so-called trellis and finds the most probable sequence by crawling the trellis based on some metric ρ. The metric can be soft and hard. The soft metric takes into account a priori probabilities and produces a soft output. The hard metric is based on *Hamming distance*. It is a number of positions where individual bits differ from input and output sequence. The target is to minimize the metric by going through the trellis.

$$\rho_{\min} = \min_{\sigma_n:\ \sigma_0 \rightarrow \sigma_{end}} \sum_i \rho_{i,d_n}(\sigma_n) \tag{2.20}$$

The best way to show the decoding of the convolutional code and to explain the Viterbi algorithm is by giving an example.

Example 2. We take the encoded sequence from Example 1. Let's suppose that the data had been sent through a noisy communication channel and we received the following message.

$$\boldsymbol{c}^T = \begin{pmatrix} 10 & 10 & 10 \end{pmatrix} \tag{2.21}$$

Before we start the decoding process we create a trellis, that is illustrated in figure 2.1

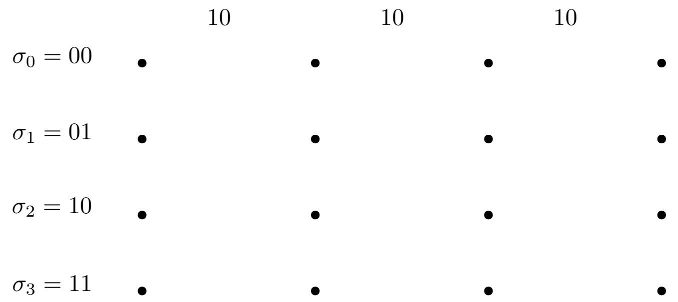


Figure 2.1: Trellis

Each point in a row represents given state $\boldsymbol{\sigma_n}$. Arrows going up denote sent symbol 0 and arrows showing down represent 1. Numbers nearby each arrow signify the expected symbol. There is also sum of Hamming distance calculated for each edge. It is the metric that is supposed to be minimized. We begin the process initially at state $\boldsymbol{\sigma}_0 = 00$.

Let's take the first state $\boldsymbol{\sigma}_0 = 00$. We create matrix $\boldsymbol{G}$

$$\boldsymbol{G} = \begin{pmatrix} \boldsymbol{G}_1 & \boldsymbol{G}_2 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}. \tag{2.22}$$

- If bit 0 was sent, it would stay in the same state $\boldsymbol{\sigma}_0$. The expected symbol is
$$\boldsymbol{c}^T_{\boldsymbol{\sigma}_0,0} = \boldsymbol{G}\boldsymbol{\sigma}_0^T = 00. \tag{2.23}$$
The Hamming distance between the expected symbol and the received one is $\rho_{1,0}(\boldsymbol{\sigma}_0) = 1$.

- If bit 1 was transmitted, it would move to the state $\boldsymbol{\sigma}_2 = 10$. The expected symbol would be
$$\boldsymbol{c}^T_{\boldsymbol{\sigma}_0,1} = \boldsymbol{G}\boldsymbol{\sigma}_2^T = 10. \tag{2.24}$$
Thus the Hamming distance is $\rho_{1,1}(\boldsymbol{\sigma}_0) = 0$.

This procedure is repeated for all states $\boldsymbol{\sigma}_n$ into which previous states expanded as well as for all received symbols until the last symbol is reached. Now we pick the node at the end of the trellis where the Hamming distance is minimal. Starting from that node we track the trellis backward choosing the edges with minimal Hamming distance.

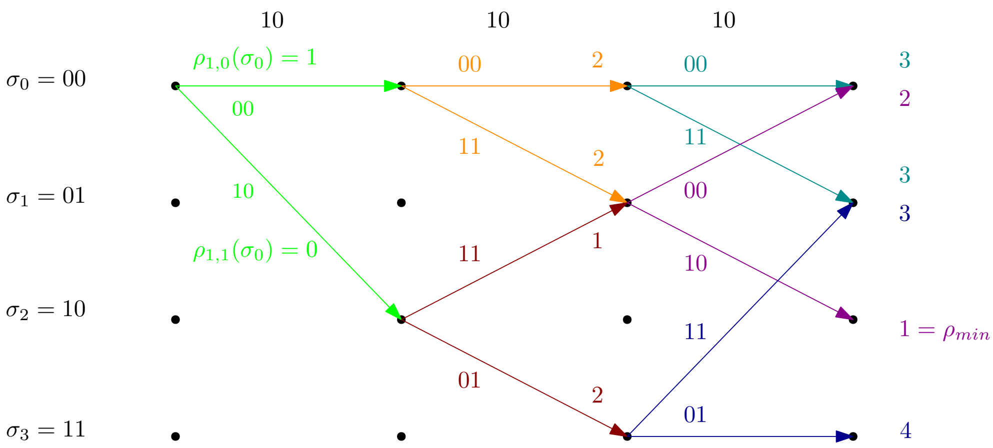


Figure 2.2: Trellis

As it can be seen from figure 2.2, the sum of hamming distances is minimal in the last column at state $\boldsymbol{\sigma_2}$. We got there from state $\boldsymbol{\sigma}_1$ by sending bit 1. From state $\boldsymbol{\sigma}_1$ the minimum leads to state $\boldsymbol{\sigma}_2$ and finally we end up at state $\boldsymbol{\sigma}_0$. The decoded message is

$$\boldsymbol{b}^T = (1\ 0\ 1). \tag{2.25}$$

Note that we received the sequence with one error bit but nevertheless we were able to correct it.

2.3 AWGN channel

To describe processes that occur in nature, we need mathematical models. One of the most common and used mathematical models of a channel is Additive White Gaussian Noise (AWGN) channel. The word *additive* signifies that the noise is simply added to a signal. Adjective *white* stands for the fact that its spectral power density $S_w(f)$ is constant across all frequencies. It is similar to white color which has uniform distribution in the visible spectrum.

$$S_w(f) = \frac{N_0}{2} \tag{2.26}$$

The normal distribution with zero mean is considered, often also called Gaussian. The input-output relation of received signal $y(t)$ and sent one $x(t)$ is

$$y(t) = x(t) + w(t), \tag{2.27}$$

or in signal space representation

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{w} \tag{2.28}$$

where $\boldsymbol{w}$ is complex, white, stacionary, rotational symmetric, zero-mean gaussian noise.
Rotational symmetric property implies that $Re(w)$ and $Im(w)$ are zero-mean Independently Identically Distributed (IID).

2.3.1 Likelihood function of AWGN channel

Let's derive the likelihood function $p(y|x)$ of AWGN channel. We will perform the derivation for scalars in signal space representation. The basic form of Gaussian distribution for real gaussian noise is

$$p_{w_r}(w_r) = \frac{1}{\sqrt{2\pi N_0}} \mathrm{e}^{-\frac{w_r^2}{2N_0}}. \tag{2.29}$$

Since $Re(w)$ and $Im(w)$ are independent we can write

$$p_w(w) = p_{w_{Re}}(w_{Re}) p_{w_{Im}}(w_{Im}) = \frac{1}{2\pi N_0} \mathrm{e}^{-\frac{w_{Re}^2}{2N_0}} \mathrm{e}^{-\frac{w_{Im}^2}{2N_0}} = \frac{1}{2\pi N_0} \mathrm{e}^{-\frac{|w|^2}{2N_0}}. \tag{2.30}$$

The knowledge of x and w gives us

$$p(y|x,w) = \delta(y - x - w) \tag{2.31}$$

Now we try to eliminate noise w from $p(y|x,w)$.

$$p(y|x) = \int_w p(y|x,w) p_w(w) \mathrm{d}w \tag{2.32}$$

$$p(y|x) = \int_w \delta(y - x - w) p_w(w) \mathrm{d}w.$$

Due to the sampling property of Dirac delta function, we get

$$p(y|x) = p_w(y - x). \tag{2.33}$$

The extension to vectors $\boldsymbol{y}$ and $\boldsymbol{x}$ is straightforward.

$$p(\boldsymbol{y}|\boldsymbol{x}) = p_w(\boldsymbol{y} - \boldsymbol{x}) = \left(\prod_k \frac{1}{2\pi N_0}\right) \mathrm{e}^{-\frac{\|\boldsymbol{y}-\boldsymbol{x}\|^2}{2N_0}} = c\ \mathrm{e}^{-\frac{\|\boldsymbol{y}-\boldsymbol{x}\|^2}{2N_0}} \tag{2.34}$$

2.3.2 Detection in AWGN channel

We assume a linear modulation with Nyquist pulses. A decoder at receiver has to be able to decide what symbol $\tilde{q}$ was received with respect to the sent symbol q_i. For that purpose, we define a metric ρ which is given to the decoder. There are various forms of metric. The most common one is the likelihood function. In this case, the decoder maximizes the likelihood function to estimate the sent symbol. It is known as ML detector (Maximum-Likelihood)

$$\tilde{q} = \arg\max_{q_i}\ p(y|q_i). \tag{2.35}$$

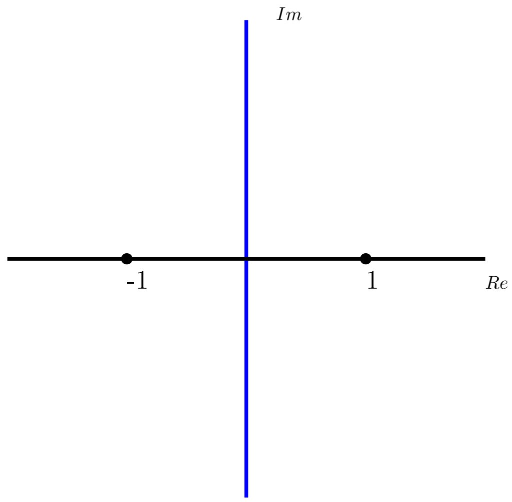
However, this calculation can be computationally demanding. Applying natural logarithm to the likelihood function seems to be a good idea. It is monotonous (not descending) function, so it does not affect the maximum nor minimum of the likelihood function.

$$\ln(p(y|x)) = -c'\|y - x\|^2 \tag{2.36}$$

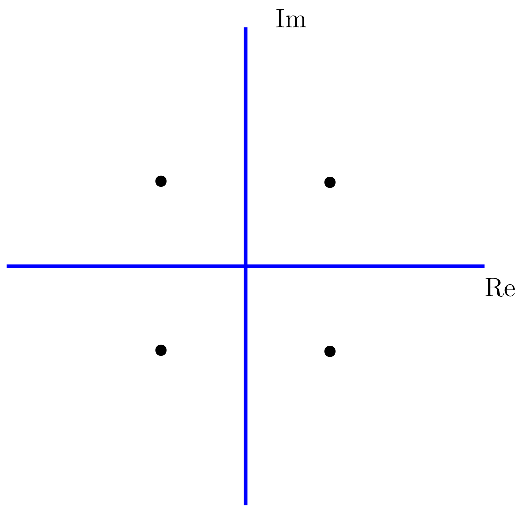
where c' is positive scaling constant not influencing the stationary points.
Now we found an equivalent metric based only on computing Euclidean distance that we need to minimize in order to get the best estimate of the sent symbol

$$\tilde{q} = \arg\min_{q_i} \|y - q_i\|^2. \tag{2.37}$$

The metric divides the constellation space into so-called decoder decision regions. Examples of decoder decision regions for BPSK and QPSK constellations are illustrated in figure 2.3. Blue lines are borders between individual regions. Thus, every signal s that falls into one of the region, is assigned to the corresponding point in constellation space.



(a) Decoder decision regions - BPSK



(b) Decoder decision regions - QPSK

Figure 2.3: Decoder decision regions

Chapter 3

Introduction to WPNC Coding

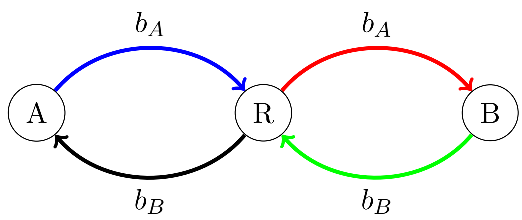
3.1 General knowledge

The idea of wireless physical layer network coding was first introduced in 2006. It uses the known fact that multiple electromagnetic waves within the same physical space are superimposed on one another.[3] In the classical way if more signals are received at the antenna of a destination node, they are treated as interference and in some applications the desired data is lost. It has been proven that using the network coding paradigm and the property of wireless channel leads to better performance and throughput.

3.1.1 Advantages of PNC

There are three approaches which can be applied in wireless networks. The main advantage of Physical-layer Network Coding (PNC) can be nicely shown in Two-way relay channel, which is the simplest topology, where PNC can be used. It is formed by two sources A and B that are also destination nodes and one relay node. We assume half-duplex constraint and binary data.

Traditional method
Both sources A and B want to communicate with each other. First, node A transmits data to the relay R. Then R retransmits the data to its destination node B. In the next phase source B transmits its data to the relay R, and R forwards it to destination node A. Each color denotes transmission at different time. It takes four time slots to complete the exchange.



Network-Layer Network Coding

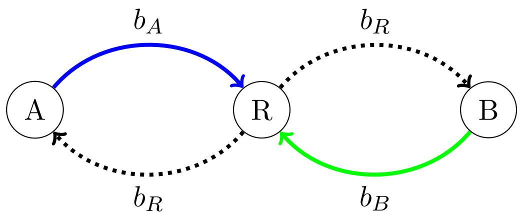
In this method the relay R computes a function (often called network code function) of data from both sources and forwards it. The destination node has to be able to unambiguously decode the desired data. Node A transmits data b_A to relay R. Then node B transmits data b_B to relay R. Now the relay can form the network code function $f(b_A, b_B)$ (exclusive OR in this case)

$$b_R = f(b_A, b_B) = b_A \oplus b_B. \tag{3.1}$$

In the last time slot, R transmits data b_R to both terminals A and B which can decode the data of interest simply by applying the exclusive OR function again

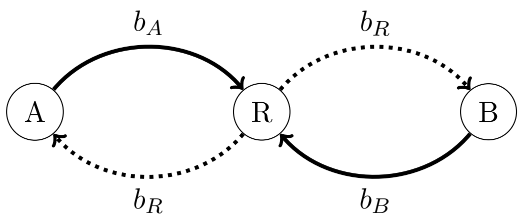
$$b_A = b_R \oplus b_B = (b_A \oplus b_B) \oplus b_B = b_A \oplus (b_B \oplus b_B) = b_A \oplus 0 \tag{3.2}$$

for node B and similary for node A. It reduces the number of time slots to three.



Physical-Layer Network Coding

Now we allow nodes A and B to transmit their data to relay R at the same time. The relay receives superimposed signal caused by the behavior of nature and extract network code function [1]. The data is no longer separable from each other. In the second time slot, R broadcasts data b_R to both destination nodes A and B. As it can be seen, this method requires only two time slots and hence it has a lot better throughput than traditional approach.



3.1.2 Relay HDF strategy and processing

Let's turn our attention to the relay. The simplest way to process the received signal at the relay is to amplify it and re-transmit. It is called *Amplify and Forward* (AF) strategy. It is promising for its simple implementation. The disadvantage of the strategy is that it also amplifies the noise. Another way often named *Joint Decode and Forward* (JDF) strategy is to decode the data at the relay from the sources separately, calculate the network code function and forward the result to the end terminals. The third strategy takes into account that the relay is not the destination point for the data. It decodes symbols created as a network code function of the sources' data. It does not know the exact messages. We refer to this processing as *Hiearchical decode and Forward* (HDF) strategy and it is the domain of our interest.

There are two stages in the relay processing we can distinguish and that cannot run simultaneously, since we assume a half-duplex constraint. In the first phase, the relay receives combined signals and it is denoted as Hierarchical Multiple Access Channel (H - MAC) stage. The adjective *hierarchical* indicates that the network topology creates a hierarchy of nodes. Each node processes some data and creates its own network code function. The decoding then depends directly on neighboring nodes. It allows us to divide the network into levels. It also implies that the network needs to be aware about its own structure. The second Hierarchical Broadcast Channel (H - BC) phase forwards the information processed by the relay. Both phases are shown in figure 3.1 on a butterfly network. This topology has two sources S_A and S_B, one relay node R and two destination nodes D_A and D_B. The goal is to transfer data b_A to D_A and data b_B to D_B.

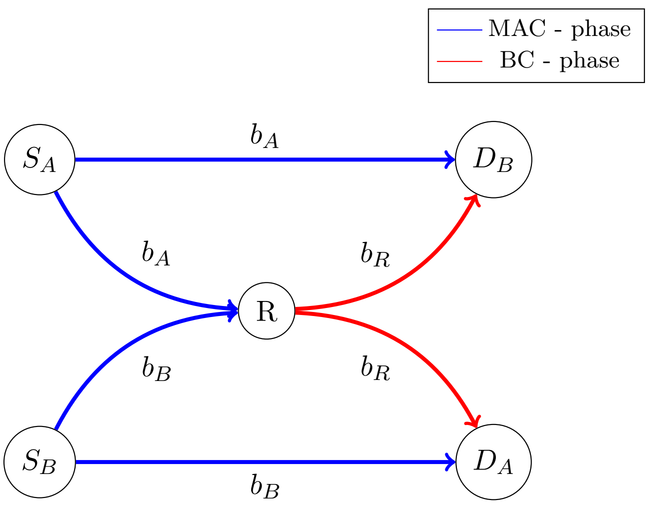


Figure 3.1: MAC and BC stage in Butterfly network

Let us define basic terminology. As it was mentioned above, the relay calculates some function f of data $\{b_i\}_{i=X,\ldots,X'}$ from multiple sources. The function will be called Hierarchical Network Code map (HNC map) and the computed symbol b is hierarchical symbol (H-symbol) [1]

$$b = f(b_X, \ldots, b_{X'}). \tag{3.3}$$

If a received signal at the relay (or destination node) carries hierarchical symbol that we wish to decode, it has useful information that we call Hierarchical Information (HI). In the butterfly network example, it is the data

b_R that is broadcasted by the relay in H-BC stage. To provide unambiguous decoding, an extra information is often required. It is called Hierarchical-Side Information (HSI) and in butterfly network, decoding the desired data is impossible without HSI at the destination node. For example the end terminal D_A needs to know, what message was sent from the source node S_B in order to decode data b_A. If the transmission of HSI is non-error, we denote it as a perfect HSI link. Note that Two-Way Relay channel is a special case of butterfly network where the HSI links between sources and destination nodes are perfect [1].

3.1.3 Isomorphic layered NCM

NCM stands for Network Coded Modulation. It is a channel coding for multiple network [1]. It takes all codebooks of neighboring nodes received in one H-MAC stage in one specific relay node and creates a new codebook for given HNC map. It needs to know the network structure. For the NCM it is easier to divide the design into two blocks. The first one will be denoted as *outer* layer and it is responsible for error correction. It is the part where we can use standard single user codes, e.g. convolutional code. The second one is called *inner* layer and it is a segment that gives the specific properties of WPNC coding. In addition, if there is isomorphism between data, code of NCM outer layer and their HNC map, we call the NCM isomorphic.

3.1.4 H-constellation

At the beginning, we have multiple sources that modulate data and send it. As it was derived earlier, if the modulation is linear and we use Nyquist pulses, the sent channel symbol corresponds to constellation point. The resulting constellation at the relay is a set of channel-combined symbols of the sources' constellation mappings. We call the set H - alphabet if any relationship between HNC map and constellation points is ignored. We define H-constellation as a set of constellation points u from the sources depending on HNC map. A subset of a particular $b = f(\tilde{b})$ is defined

$$U(b) = \{u;\ u = u(\{s_i(b_i)\}_{i\in S}), \tilde{h} | b = f(\tilde{b})\}, \tag{3.4}$$

where $\tilde{h}$ is a set of channel parametrizations, $f(\tilde{b})$ denotes HNC map and $\tilde{b} = \{b_i\}_{i\in S}$. [[1], page 64] Then, H-constellation is union of all subsets depending on particular b

$$U = \bigcup_{\forall b\ :\ b=f(\tilde{b})} U(b). \tag{3.5}$$

3.1.5 Hierarchical demodulator in AWGN channel

The relay in H-MAC stage receives signal x_n. We need some function that can decide what symbol was sent based on a received signal x_n. As it was mentioned in chapter 2, this function is called metric and it is given to the decoder. We assume AWGN memoryless channel

$$x_n = u_n(\tilde{b}_n) + w_n \tag{3.6}$$

where w_n is gaussian complex-valued noise with variance σ_w^2 and $u_n(\tilde{b}_n)$ is a channel-combined symbol and also an element of H-alphabet.
It can be shown [[1], page 85] that the likelihood function (for 1-dimensional constellation symbols) looks as follows

$$p(x_n|b_n) = \frac{1}{2\pi\sigma_w^2} \sum_{\tilde{b}_n : f(\tilde{b}_n) = b_n} \exp\left(-\frac{1}{\sigma_w^2}\|x_n - u_n(\tilde{b}_n)\|^2\right). \tag{3.7}$$

Unfortunately, we cannot simplify the metric as we did in decoding point-to-point scenario. However, the metric can be approximated. We take only the dominant exponential in the equation 3.7. That is the exponential with minimum argument. Thus, the likelihood function reduces to single exponential. Now it is equivalent to the case we had before in classical decoding in AWGN channel. The metric becomes Euclidean distance (H - distance) [1]

$$\rho^2_{Hmin}(x_n, b_n) = \min_{\tilde{b}_n : f(\tilde{b}_n) = b_n} \|x_n - u_n(\tilde{b}_n)\|^2. \tag{3.8}$$

3.1.6 Channel parametrization

Depending on the number of transmitted signals, the channel has several degrees of freedom (parameters) that can drastically change the shape of H-constellation. We will investigate the impact on H-constellation after two signals go through a channel and they are superimposed. Let s_A be a transmitted signal of source S_A and s_B a transmitted symbol of source S_B in signal space representation. The input-output relation is

$$u = h_A s_A + h_B s_B \tag{3.9}$$

where u denotes received signal in signal space, h_A and h_B are complex-valued coefficients.
If we consider $h_A \neq 0$, the equation 3.9 can be rewritten

$$u = h_A\left(s_A + \frac{h_B}{h_A} s_B\right) = h_A(s_A + h s_B). \tag{3.10}$$

The coefficient h_A is called common (linear) fading and h is relative fading. Common fading h_A is the one that only rotates and sets the amplitude of the H-constellation and it is an issue which also appears in classical point-to-point communication. It can be corrected by standard synchronization methods. We set the parameter h_A to 1. The specific problem in WPNC networks and especially in HNC map design is the coefficient h.

Chapter 4

WPNC Coding in three-source-node network

In this chapter, we will focus on three-source-node network topology. The impact on H-constellation at the relay node will be studied as well as the quality of HNC map depending on channel parametrization.

4.1 System model

Let us suppose a topology with sources S_A, S_B and S_C. All sources send binary data and use BPSK modulation. There is also a relay node R. Channels between the sources and the relay node are not orthogonal and hence the relay node receives channel-combined signal u from the sources. We will suppose that the HNC map at the relay has cardinality $|A_H| = 4$. The input-output model of the channel is

$$u = h_A s_A + h_B s_B + h_C s_C. \tag{4.1}$$

As we did before, under the assumption $h_A \neq 0$ we can rewrite the equation 4.1 to the following form

$$u = h_A(s_A + \frac{h_B}{h_A} s_B + \frac{h_C}{h_A} s_C) = h_A(s_A + h_1 s_B + h_2 s_C), \tag{4.2}$$

where s_A, s_B, s_C are signals transmitting sources S_A, S_B and S_C respectively, h_1 and h_2 are complex-valued relative fadings. Moduls $|h_1| = |h_2| = 1$ are considered. h_A is set to one. The topology is shown in figure 4.1.

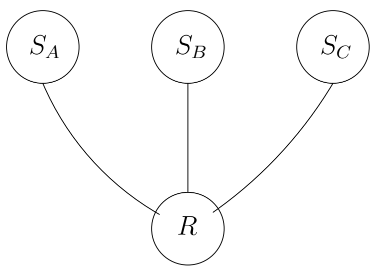
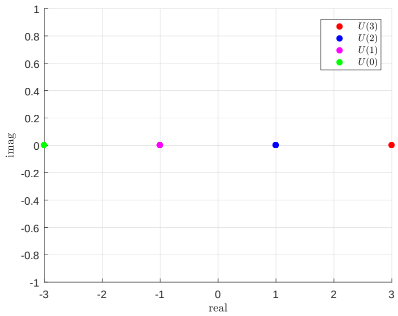


Figure 4.1: Three-source-node network Topology

4.2 H-constellation

The H-constellation depends on the HNC map. The problem is that there are two parameteres h_1 and h_2 that are non-linear and change the shape of H-aphabet. The HNC maps are defined by tables. The first row of the table signifies sent symbol of source S_B. In the first column, there are symbols sent by source S_A and the last row indicates sent symbols of source S_C. In figure 4.2, we can see the H-constellation and its corresponding HNC map for both relative fadings set to 1. Thus, there is one combination of sent symbols that is assigned to H-symbol $b = f_1(s_A, s_A, s_C) = f_1(-1, -1, -1) = 0$, three combinations that are transformed to $b = 1$, another three combinations for $b = 2$ and finally one combination where f computes $b = 3$. If the relative fadings change, the HNC map could no longer be sufficient and there may appear points that are very close to each other (they can even be in the same position) but they belong to another H-symbol. We call these points singular fadings. Singular fadings must be assigned to the same H-symbol. We now show H-constellation and its proper HNC map for different channel parametrization.

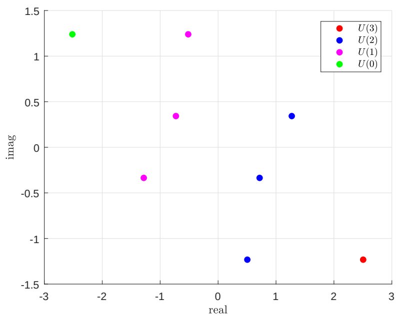


(a) H-constellation with $h_1 = 1$ and $h_2 = 1$

s_A \ s_B	-1	1	-1	1
-1	0	2	1	1
1	1	3	2	2
s_C	-1	1	1	-1

(b) HNC map f_1

Figure 4.2: HNC map f_1 and H-constellation

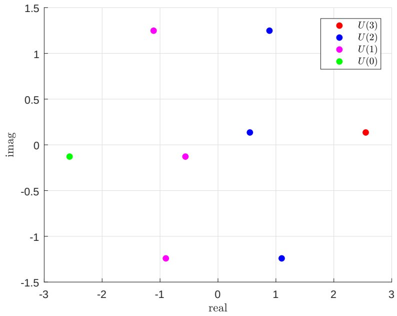


(a) H-constellation with $h_1 = e^{j\pi\frac{85}{10}}$ and $h_2 = e^{j\pi\frac{71}{10}}$

s_A \ s_B	-1	1	-1	1
-1	2	0	1	1
1	3	1	2	2
s_C	-1	1	1	-1

(b) HNC map f_2

Figure 4.3: HNC map f_2 and H-constellation

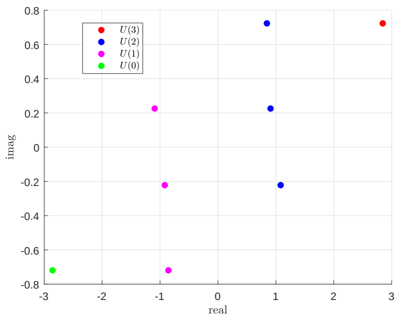


(a) H-constellation with $h_1 = \mathrm{e}^{j\pi \frac{81}{10}}$ and $h_2 = \mathrm{e}^{j\pi \frac{12}{5}}$

s_A \ s_B	-1	1	-1	1
-1	1	1	2	0
1	2	2	3	1
s_C	-1	1	1	-1

(b) HNC map f_3

Figure 4.4: HNC map f_3 and H-constellation



(a) H-constellation with $h_1 = \mathrm{e}^{j\pi \frac{15}{10}}$ and $h_2 = \mathrm{e}^{j\pi \frac{4}{5}}$

s_A \ s_B	-1	1	-1	1
-1	1	1	0	2
1	2	2	1	3
s_C	-1	1	1	-1

(b) HNC map f_4

Figure 4.5: HNC map f_4 and H-constellation

For curiosity, figure 4.6 shows the fact when HNC map would have poor properties because different H-symbols are close to each other.

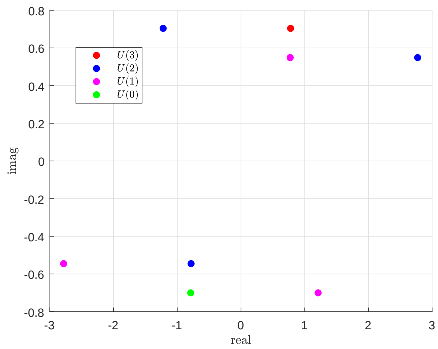


Figure 4.6: Failure of HNC map

4.3 Quality of HNC map

We have a couple of HNC maps. There comes up another question. The usability of the HNC maps is limited by the relative fadings. The HNC maps were designed for specific channel parametrization. We would like to have a criterion that would allow us to decide in which range of values of relative fadings we can utilize given HNC map. In this thesis, we come up with two ad-hoc approaches. Both approaches work with hierarchical distance (H - distance). [1]

Definition. Let $u(\tilde{b})$ be a channel-combined symbol depending on sent symbols $\tilde{b}$, f is given HNC map, $f(\tilde{b}_j) = b$, $f(\tilde{b}'_i) = b'$ are two different H-symbols where $\tilde{b}_j \neq \tilde{b}'_i$. We call $d_{j,i}(b, b')$ H - distance if and only if

$$d_{j,i}(b, b') = \|u(\tilde{b}_j) - u(\tilde{b_i}')\|^2. \tag{4.3}$$

The H-distance expresses the Euclidean distance between two different H-symbols in H-constellation. Note that there can be several H-distances between two H-symbols since there can be several combinations of sent symbols assigned to the same H-symbol.

Example. Let's take the H-constellation from figure 4.4. We take a look at $d_{j,i}(0, 1)$. H-symbol $b = 0$ is obtained by applying the HNC map to sent symbols $\tilde{b}_1 = s_A, s_B, s_C = -1, 1, 1$. Thus, the index $j = 1$. The other one $b' = 1$ is gotten by applying HNC map to symbols $\tilde{b_1}' = s_A, s_B, s_C = 1, 1, 1$, $\tilde{b_2}' = -1, -1, 1$ and $\tilde{b_3}' = -1, 1, -1$, hence $i \in \{1, 2, 3\}$. Thus,

$$d_{1,1}(0, 1) = \|u(\tilde{b}_1) - u(\tilde{b_1}')\|^2 = 2^2 = 4 \tag{4.4}$$
$$d_{1,2}(0, 1) = \|u(\tilde{b}_1) - u(\tilde{b_2}')\|^2 = 2^2 = 4 \tag{4.5}$$
$$d_{1,3}(0, 1) = \|u(\tilde{b}_1) - u(\tilde{b_3}')\|^2 = 2^2 = 4 \tag{4.6}$$

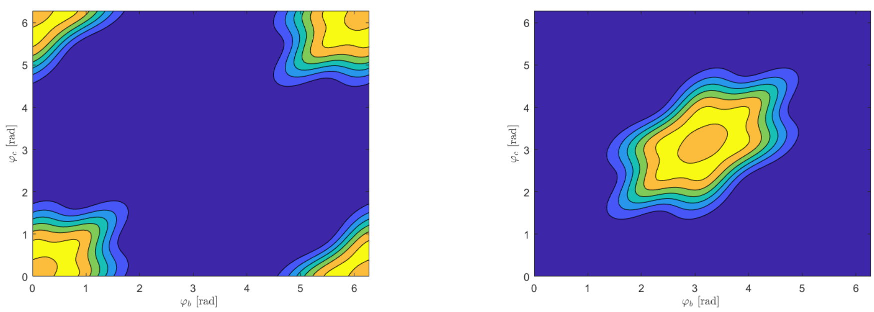
In this case, the H-distance is the same for all channel-combined symbols that belong to the H-symbols $b = 0$ and $b' = 1$. It is just a coincidence. The values of distances can differ. We denote a set $C(b) = \{c : 1, 2, \ldots, |U(b)|\}$.It is obvious that for $d_{j,i}(1, 2)$, there are nine H-distances that can be evaluated (i.e. $j \in C(1), C(1) = \{1, 2, 3\}$ and $i \in C(2), C(2) = \{1, 2, 3\}$).

4.3.1 Product performance metric

The first approach is based on computing a product of all H-distances $d_{j,i}(b, b')$ between different H-symbols. The idea is simple. If there are two different H-symbols in H-constellation very close to each other they have small H-distance $d_{j,i}(b, b')$. Thus, it decreases the product to zero and it suggests that the HNC map in this area is useless because a singular fading is created. Specifically in our case, we basically compute a function

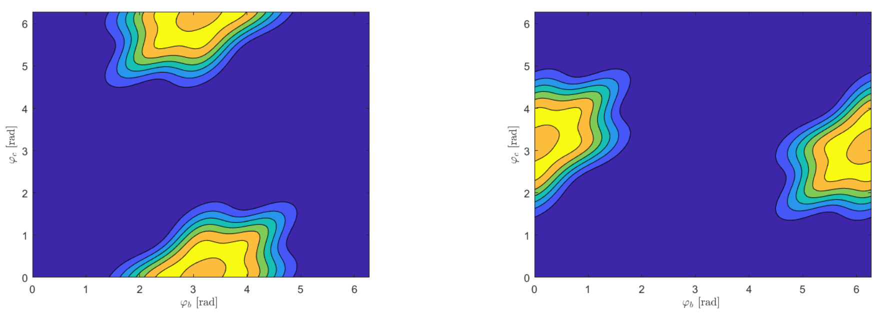
$$\kappa = \prod_{b=0}^{3} \prod_{b'>b}^{3} \prod_{\forall j \in C(b)} \prod_{\forall i \in C(b')} d_{j,i}(b, b'). \tag{4.7}$$

The function is implicitly dependent on channel parametrization. The following pictures show the results for individual HNC maps. The blue colour signifies values close to zero and hence the designed HNC map should not be used in this area. Conversely, the yellow colour should be the best option for given HNC map. It nicely illustrates the areas where each HNC map can be applied.



(a) quality of HNC map f_1 (b) quality of HNC map f_2

Figure 4.7: Quality of HNC maps f_1 and f_2



(a) quality of HNC map f_3 (b) quality of HNC map f_4

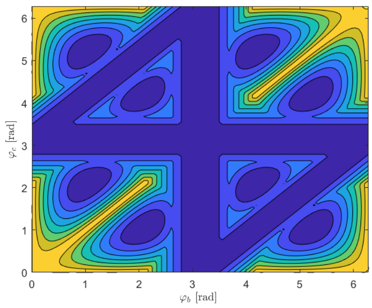
Figure 4.8: Quality of HNC maps f_3 and f_4

4.3.2 Min H-distance perfarmance metric

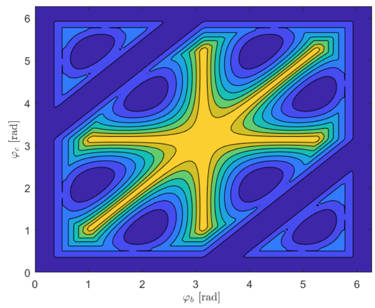
The second approach also takes into account the H-distance. We take a minimum of all possible H-distances depending on channel parametrization. We compute a function that looks as follows

$$\kappa_{min} = \min_{\forall b, \;\; \forall b': b' > b, \;\; \forall j \in C(b), \;\; \forall i \in C(b')} d_{j,i}(b, b') \tag{4.8}$$

This function is calculated for all channel parametrization with moduls set to one. We can see that the graphs are slightly different.

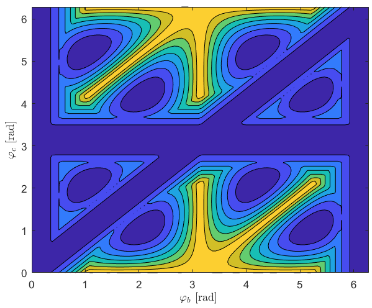


(a) quality of HNC map f_1

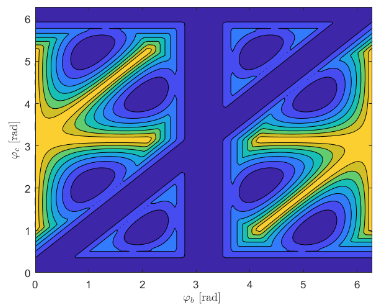


(b) quality of HNC map f_2

Figure 4.9: Quality of HNC maps



(a) quality of HNC map f_3



(b) quality of HNC map f_4

Figure 4.10: Quality of HNC maps

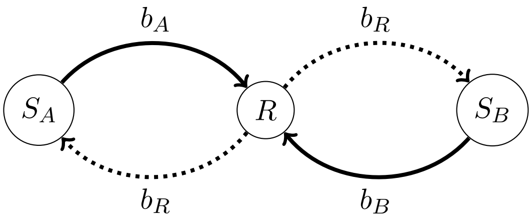
Chapter 5

Simulation and results

In this chapter, we are going to implement WPNC coding in Two-Way-Relay channel. We take a look at convolutional encoder and decoder implementation. Different constellations for sources will be used as well as encoded and uncoded case.

5.1 System model

For the purpose of the simulation, we suppose two-way relay channel topology and half-duplex constraint. We use isomorphic layered Network Coded Modulation (NCM) and the relay performs HDF processing strategy. The sources send binary data. The convolutional code is implemented for encoded transmission. The convolutional decoder uses hard decision metric (i.e. Hamming distance). Our target is to transfer data b_A to its destination node S_B and similarly data b_B to its destination node S_A and compute the end-to-end error rate performance.



5.2 Convolutional encoder implementation

We will design a convolutional encoder which will be used in the simulation later on. It is convenient to implement the encoder as a finite state machine with states $\boldsymbol{\sigma}_n$. We assume binary data. The generator matrices will be

$$\boldsymbol{G}_0 = \begin{pmatrix}1\\0\\0\\1\end{pmatrix}, \ \boldsymbol{G}_1 = \begin{pmatrix}1\\1\\1\\0\end{pmatrix}, \ \boldsymbol{G}_2 = \begin{pmatrix}0\\0\\1\\1\end{pmatrix}, \ \boldsymbol{G}_3 = \begin{pmatrix}1\\1\\1\\1\end{pmatrix}. \tag{5.1}$$

For convenience, the generator matrices will be joint to one generator matrix

$$\boldsymbol{G} = \begin{pmatrix} \boldsymbol{G}_0 & \boldsymbol{G}_1 & \boldsymbol{G}_2 & \boldsymbol{G}_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}. \tag{5.2}$$

The number of matrices $\boldsymbol{G}_i$ (columns of $\boldsymbol{G}$) indicates the number of shift registers and hence there are 4 memory registers, all representing one bit and set to 0. The positions of registers are shown in figure 5.1.

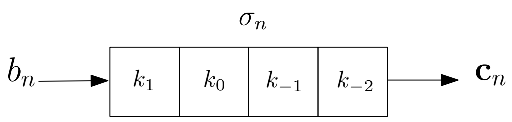


Figure 5.1: Shift registers

If new bit enters in k_1, all bits move to the right and the last one, which is in register k_{-2}, is thrown away. From this new state, an encoded symbol is calculated using finite (Galois) field arithmetic and the process is repeated. The convolutional encoder is implemented in Matlab function *conv_code.m*. The function takes `data` and the generator matrix `G` and computes the desired encoded message. It is possible to set arbitrary generator matrix, but for the Wireless Physical-layer Network Coding (WPNC) simulation, the generator matrix mentioned above is used. Input data is given to the function in matrix `data` $\in M^{n \times m}$. Columns of the matrix are the data sequence. Rows are data that entres the registers and replaces m first positions. m is called prefix. `memory` stands for the number of shift registers (i.e.number of columns in `G`).

```
function encoded = conv_code(data,G)
    m = size(data,2);
    memory = size(G,2); %number of shift registers
    %allocation
    state = zeros(memory,1);
    encoded = zeros(memory*m,1); %encoded message
    b = 1;
    for k = 1:size(data,1)
        word = data(k,:);
        state = [word'; state(1:end-m)];
        encoded(b:b+size(G,1)-1) = (mod(G*state,2));
     b = b+size(G,1);
    end
end
```

To give an example of our convolutional encoder, we can take a look at figure 5.2. There is state $\boldsymbol{\sigma}_n = (1\ 0\ 1\ 0)$. To obtain the first encoded bit, the first row in matrix G gives us clear instructions to add up bits in registers k_1, k_0 and k_{-2}. The second bit is computed with respect to the second row in matrix G by adding up bits in position k_0 and k_{-2}. This procedure is also done for the last two bits. The resulting encoded symbol is

$$\boldsymbol{c}_n^T = (1\ 0\ 1\ 0). \tag{5.3}$$

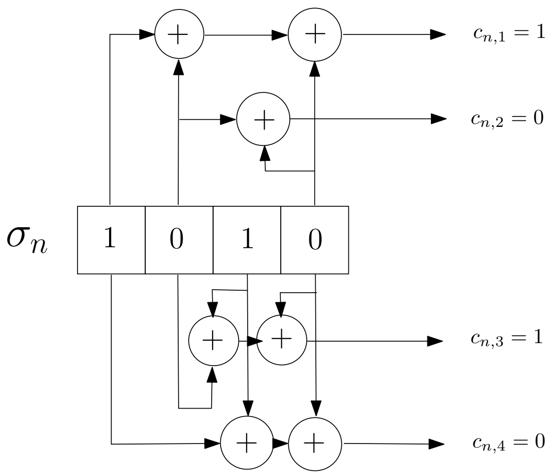


Figure 5.2: Example of encoding

We can get the same result simply by utilizing matrix multiplication (this is why $\boldsymbol{G}$ is convinient). We can multiply generator matrix $\boldsymbol{G}$ with the state $\boldsymbol{\sigma}_n$.

$$\boldsymbol{c}_n = \boldsymbol{G}\boldsymbol{\sigma}_n^T = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \tag{5.4}$$

5.3 Convolutional decoder implementation

The decoding process uses Viterbi algorithm and it is split into two Matlab functions *generate_trellis*, which consists of another *expand_node* function, and *decode*. The first function *generate_trellis* takes received encoded data `Encoded`, the generator matrix `G` and prefix `m`. The decoder expects that the first encoded symbol was calculated with all registers set to 0.

1. 3-Dimensional matrix (`trellis`) is created. Since code rate

   $$R = \frac{1}{\texttt{size(G,1)}} = \frac{\texttt{message\_length}}{\texttt{length(Encoded)}}$$

   we can easily calculate the `message_length`.

```
message_length = length(Encoded)/size(G,1);
memory = size(G,2);
trellis = NaN(2^memory,message_length,2);
```

2. The sumation of hamming distace is stored in `trellis(:,:,1)` and the state $\boldsymbol{\sigma}_n$ from which we get to a new state $\boldsymbol{\sigma}_{n+1}$ is stored in `trellis(:,:,2)`.

3. In the first iteration the function *expand_node* takes column `trellis(:,1,:)`, current sum of hamming distance `ham_dist_be`, generator matrix `G`, `state`, received `word` and prefix `m` and expands it to new two states.

```
trellis(:,1,:) = ...
    expand_node(trellis(:,1,:),0,G,0,word,memory,m);
```

4. In *expand_node* function, all possible combinations of states (`newstate`) are created depending on the received symbol (`bin_l`). For each state, expected word is computed as well as hamming distance between the expected word (`new_word`) and a received one (`word`). It passes through individual states. If the state wasn't expanded or the sum of hamming distance is higher the data are rewritten.

```
function part_trellis = ...
    expand_node(trellis_col,ham_dist_be,G,state, ...
    word,memory, m)
   part_trellis = trellis_col; %trellis column
   for l = 1:2^m
       bin_l = de2bi(l-1,m); %possible received symbol
       %current state in binary system
       bin_state = de2bi(state,memory);
       %new state in binary system depending on bin_l
       newstate = [bin_l bin_state(1:end-m)];
       new_word = mod(G*newstate',2); %expected word
       hamming_dist = sum(new_word ≠ word);

       if  isnan(trellis_col(bi2de(newstate)+1,1)) ||  ...
           trellis_col(bi2de(newstate)+1,1)≥ ...
           hamming_dist+ham_dist_be
           %new sum of hamming distance
           part_trellis(bi2de(newstate)+1,1) = ...
               ham_dist_be+ hamming_dist;
           part_trellis(bi2de(newstate)+1,2) = state;
       end
   end
```

5. In the next iteration (`g` signifies the current iteration number) it takes another column. It finds states which were written to the trellis in previous iteration and expands them again.

```
for x = 1:2^memory
  if ¬isnan(trellis(x,g-1,1))
     state = x-1;
     trellis(:,g,:) = ...
         expand_node(trellis(:,g,:),trellis(x,g-1,1),...
                    G,state,word,memory,m);
  end
end
```

6. This is repeated until the end of the matrix (`g = message_length`) is reached. The function returns completed trellis.

Then the second function *decode* is called. It requires the computed trellis and prefix.

1. It finds the least sum of hamming distance ρ_{min} in the last column.

```
%message length
N = size(trellis,2);
%the least sum of hamming distance in the last column
[¬,state] = min(trellis(:,N,1));
```

2. The target state of the least sum of ρ_{min} is found and the first m (prefix) bits is the last symbol of the original message `data`.

```
%number of shift registers
len = log2(size(trellis,1));
state = state-1;
%first m bits of the state are stored as the received ...
    symbol
binary = de2bi(state,len);
data(N, :) = binary(1:m);
```

3. Then it backtracks to the state from which the current state was expended. This is repeated until state $\boldsymbol{\sigma}_0$ is reached.

```
for n = N:-1:2
    state = trellis(state+1,n,2);
    binary = de2bi(state,len);
    data(n-1, :) = binary(1:m);
end
```

5.4 Two-Way-Relay channel - BPSK

We have two BPSK sources, both sending binary data. In MAC stage the sources send signals $s_A \in \{-1, 1\}$ and $s_B \in \{-1, 1\}$ depending on the data. The task is to find HNC map performed by relay node so that no information is lost. Before transmission, we can encode the data with convolutional code and the process will not change due to the isomorphic layered NCM. We assume AWGN channel

$$x_n = s_{A,n} + hs_{B,n} + w_n. \tag{5.5}$$

The H-constellation at the relay is influenced by the channel parametrization h. If the decoder does not know the relative fading, error rate can be high. We suppose that the relay knows the parameter h. We will mark the received data as a sequence $c_n = b_{A,n}\ b_{B,n}$ (e.g. c = 01 if source S_A sent $b_A = 0$ and source S_B sent $b_B = 1$). It is necessary to realize that due to the HDF strategy, the relay does not need to know the sent data. It only needs to know the hierarchical symbol. However, it is done for the sake of clarity.

b_A / b_B	0	1
0	0	1
1	1	0

Table 5.1: Hierarcical function f - XOR

Figure 5.3 shows that for $h = 1$ there are two combinations of sent data that belong to one hierarchical constellation point. In order to be able to unambiguously decode the data, the HNC map has to assign these two combinations in one hierarchical symbol. Then we say that a singular fading is resolved. It can be shown that the only HNC map possible in this example is XOR function. It is defined in table 5.1.

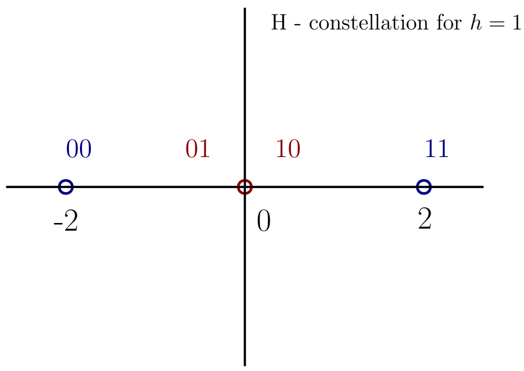
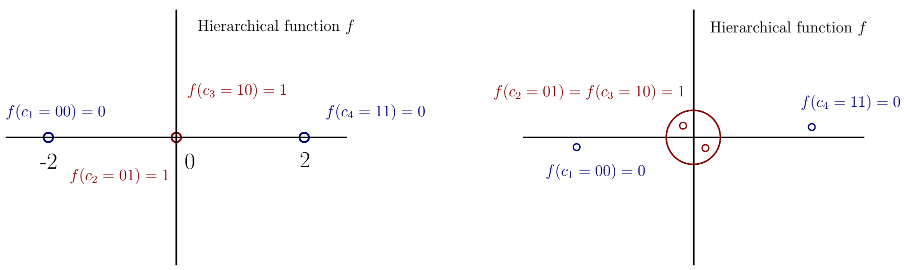


Figure 5.3: H - constellation

Figure 5.4 illustrates how the relative fading h can change the shape of H-constellation. The same rules apply for the HNC map. The points around the origin of the coordinate system have to be mapped to the same symbol.



(a) H - constellation with $h = 1$

(b) H - constellation with $h = 0.8e^{j\frac{\pi}{10}}$

Figure 5.4: H - constellations with different h

The simulation uses two types of detection metric at the relay decoder. The first one is the real metric that depends on the noise variance σ_w^2.

$$p(x_n|0) = \frac{1}{2\pi\sigma_w^2}\left(\exp\left(-\frac{1}{\sigma_w^2}||x_n - u(c_1)||^2\right) + \exp\left(-\frac{1}{\sigma_w^2}||x_n - u(c_4)||^2\right)\right)$$
$$p(x_n|1) = \frac{1}{2\pi\sigma_w^2}\left(\exp\left(-\frac{1}{\sigma_w^2}||x_n - u(c_3)||^2\right) + \exp\left(-\frac{1}{\sigma_w^2}||x_n - u(c_2)||^2\right)\right)$$

where $u(c_i)$ denotes channel combined constellation point and x_n is received signal. We choose the hierarchical symbol $b_R \in \{0,1\}$, which has higher probability for the received symbol x_n. This process is called Maximum-Likelihood (ML) detection.

$$\tilde{b}_R = \arg\max_{b_R} p(x_n|b_R) \tag{5.6}$$

The decoder decision regions for relative fading $h = 0.9\exp(j\frac{\pi}{4})$ are shown in figure 5.5

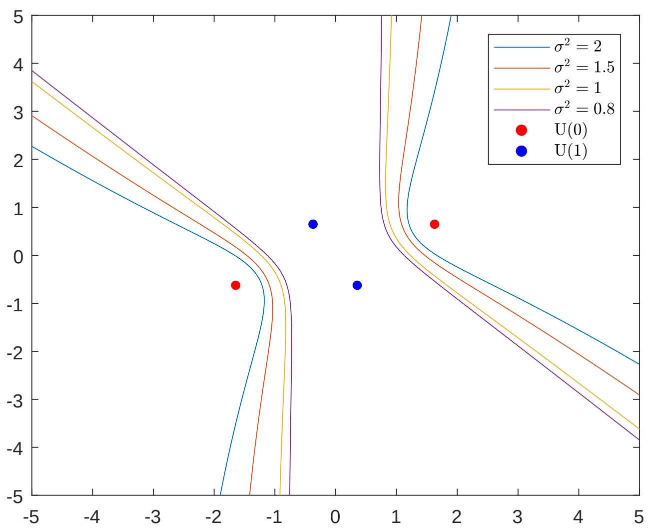


Figure 5.5: Real decoder regions for uncoded transmission, $h = 0.9\exp(j\frac{\pi}{4})$

The second metric is only an approximation of the real metric based on Euclidean distance.

$$\rho^2_{Hmin}(x_n, 0) = \min\{||x_n - u(c_1)||^2, ||x_n - u(c_4)||^2\} \tag{5.7}$$
$$\rho^2_{Hmin}(x_n, 1) = \min\{||x_n - u(c_2)||^2, ||x_n - u(c_3)||^2\} \tag{5.8}$$

The decision is then done by the equation

$$\tilde{b}_R = \arg\min_{b_R} \rho^2_{Hmin}(x_n, b_R) \tag{5.9}$$

The decoder regions are visualized in figure 5.6

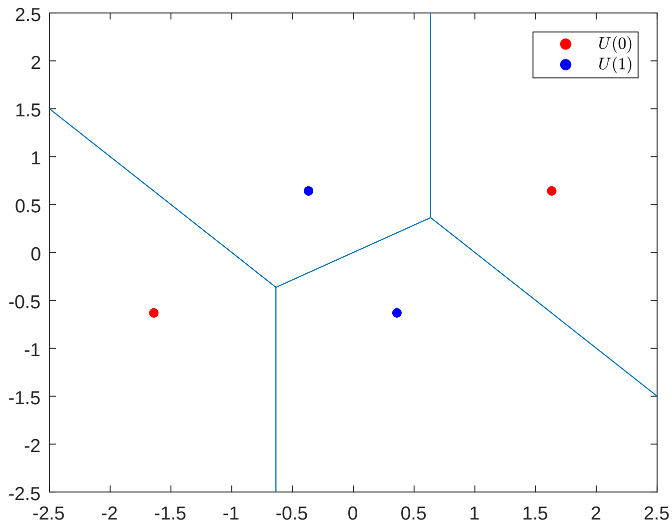


Figure 5.6: Approximeted decoder decision regions for $h = 0.9\exp(j\frac{\pi}{4})$

In BC stage, after the hierarchical symbols are detected, they are again assigned to BPSK constellation points s_n and broadcasted through AWGN channel.

$$x_R = s_R + w \tag{5.10}$$

Classical point-to-point detection in AWGN channel is used based on Euclidean distance as it was derived in section 2.3. Symbol -1 is retransformed to data 0 and symbol 1 to data 1. For BPSK constellation, it is simply computed by the equation 5.11 since the decision region is divided by the imaginary axis to two areas .

$$\tilde{b}'_R = \frac{1}{2}\left[\text{sign}(Re(x_R)) + 1\right] \tag{5.11}$$

Then, to obtain the desired data, XOR function is applied to received symbol $\tilde{b}'_R$ and sent data b_i.

$$\tilde{b}_A = \tilde{b}'_R \oplus b_B, \tag{5.12}$$
$$\tilde{b}_B = \tilde{b}'_R \oplus b_A \tag{5.13}$$

If we encoded the data at the beginning, now we use Viterbi algorithm to decode it. The error-rate performence is calculated and the simulation is over.

5.5 Two-Way-Relay channel - QPSK

In the second simulation, we use QPSK sources transmitting signals $s_B = s_A = \{e^{j\pi\frac{(2i-1)}{4}}\}_{i=1}^{4}$ designed according to the figure 5.7. For the sake of simplicity, the data is converted to decimal system. The sent data is expressed as a sequence $c_n = b_{A,n}\, b_{B,n}$.

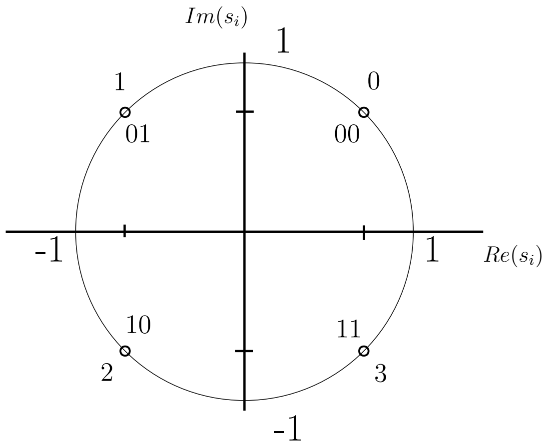
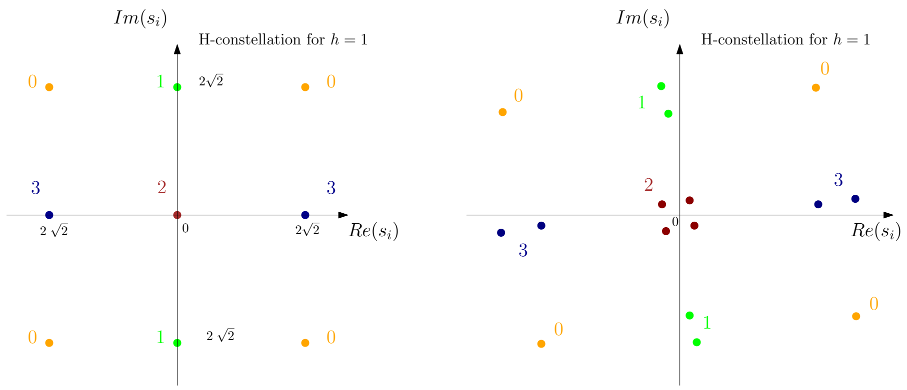


Figure 5.7: QPSK constellation

As in the previous case, we can encode the data by convolutional code. Messages are sent through AWGN channel according to equation 5.5. The H-constellation at the relay is shown in figure 5.8.



(a) H - constellation with $h = 1$ (b) H - constellation with $h = \exp^{j\frac{\pi}{10}}$

Figure 5.8: H - constellation for different h

For all points that fall into the same constellation point, the HNC function has to compute the same network code symbol. There are more functions for given channel parametrization h that fulfill conditions of HNC map so that it resolves all the singular fadings. We define the HNC map by the table 5.2. It can be recognized as bit-wise XOR function. The figure 5.8 shows the fact that all clashes are resolved. The detection at the relay is computed based on

b_A / b_B	0	1	2	3
0	0	1	2	3
1	1	0	3	2
2	2	3	0	1
3	3	2	1	0

Table 5.2: Hierarchical function f

two metrics. The real one calculates likelihood probability according to 3.7. The second approximated metric is based on the Euclidean distance between received signal x_n and H-symbols in H-constellation. Our goal is to minimize it. Both decoder regions and H-constellation are visualized in figure 5.9.

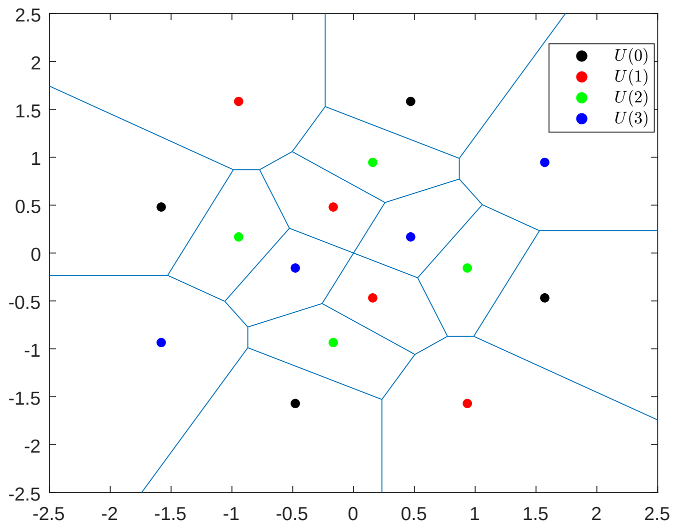


Figure 5.9: Decoder regions and H - constellation for $h = 0.9e^{j\frac{\pi}{3}}$

After the HNC map is computed, the hierarchical symbols are converted to QPSK constellation points according to figure 5.7 and again broadcasted through AWGN channel. For classical QPSK in AWGN channel, the decoder decision regions are divided by the angle in constellation space.

$$\forall \arg(x_R) \in \left(0, \frac{\pi}{2}\right\rangle \rightarrow \quad b_R = 00 \tag{5.14}$$

$$\forall \arg(x_R) \in \left(\frac{\pi}{2}, \pi\right\rangle \rightarrow \quad b_R = 01 \tag{5.15}$$

$$\forall \arg(x_R) \in \left(\pi, \frac{3\pi}{2}\right\rangle \rightarrow \quad b_R = 10 \tag{5.16}$$

$$\forall \arg(x_R) \in \left(\frac{3\pi}{2}, 2\pi\right\rangle \rightarrow \quad b_R = 11 \tag{5.17}$$

We apply XOR function to the received data and data from the source (eq. 5.12 and 5.13). In case of encoded data transmission by convolutional code, we use Viterbi algorithm to decode it. Finally, the error-rate is calculated.

5.6 Results

The simulations are implemented in MATLAB. Because of the presence of the noise, plots will change for each simulation. It is possible to set any required length of the data sequence.

5.6.1 TWRC - BPSK

Implementation of TWRC with BPSK sources can be found in MATLAB script *BN_euclidean_metric_BPSK* and *BN_real_metric_BPSK*. Data is created in script *data*.
First, we simulate the trasmission for both real and approximated metric. For the purposes of the simulation, length of 10^5 symbols was chosen. The relative attenuation was set to $h = 0.9\exp(j\frac{\pi}{4})$ and also to $h = 0.6\exp(j\frac{\pi}{5})$. As it can be seen from figure 5.10, it really does not matter if we use the real or approximated metric. The efficiency of the approximation is very high in this case. On the other hand, symmetry of the channel has a big influence on the overall performance.

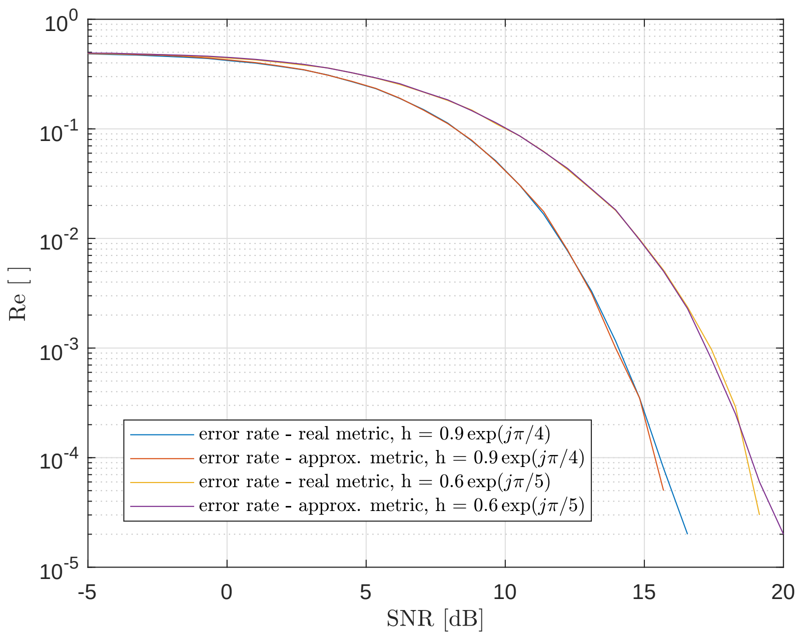


Figure 5.10: Simulation for both real and approximated metric - BPSK

Now let's investigate the impact on the bit-error rate by using the convolutional code. Length of 10^4 bits was set to the simulation. We use the Euclidean metric. As the picture 5.11 shows, the usage of covolutional code significantly influences the performance. It clearly satisfies the purpose for which it was created.

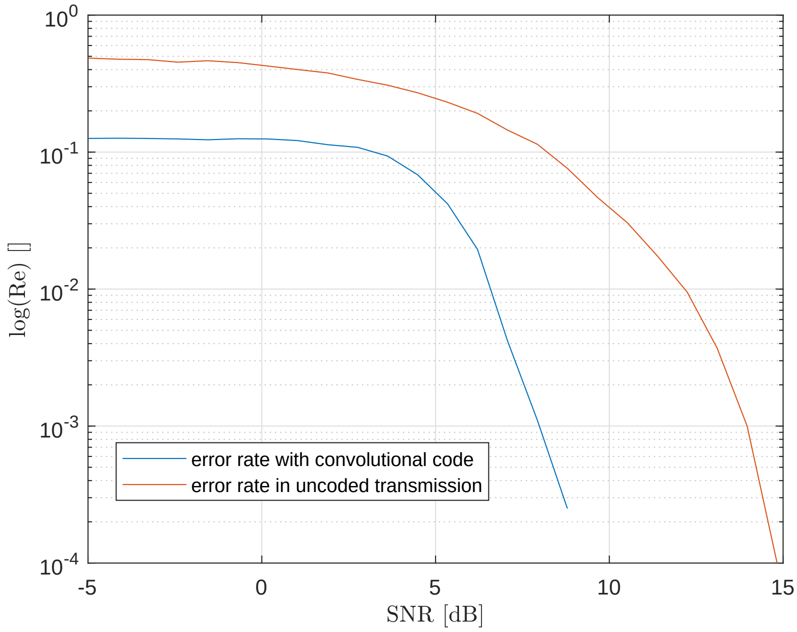


Figure 5.11: Simulation of coded and uncoded case, euclidean metric with $h = 0.9\exp(j\frac{\pi}{4})$

5.6.2 TWRC - QPSK

Similarly, the error rate performance for QPSK sources was simulated in Matlab functions *BN_real_metric_QPSK* and *BN_real_metric_QPSK*. The similar simulations were run as in previous subsection 5.6.1. The first one compares the differences in use of different metrics and also the impact of channel parametrization. It is illustrated in figure 5.12 The second one focuses on the usage of convolutional code but now the real metric is used. It can be seen in figure 5.13.

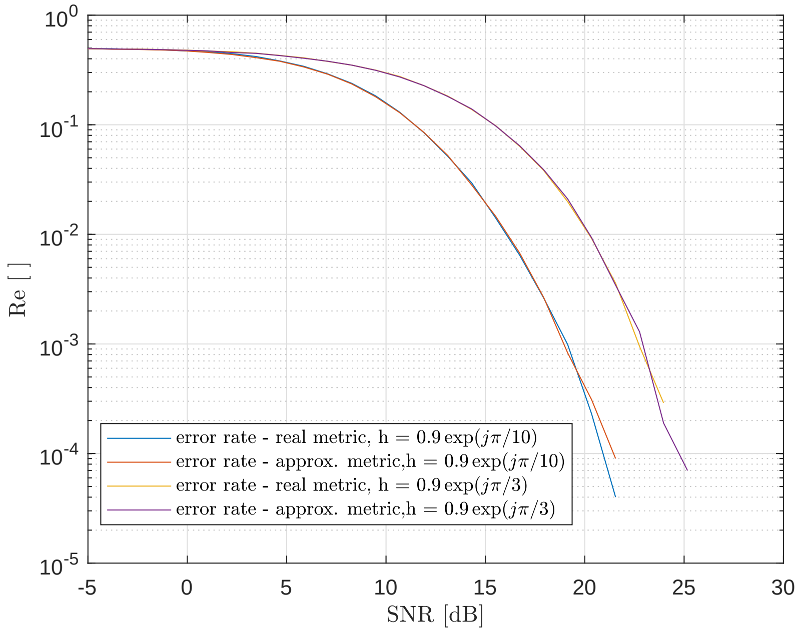


Figure 5.12: Simulation for both real and approximated metric - QPSK

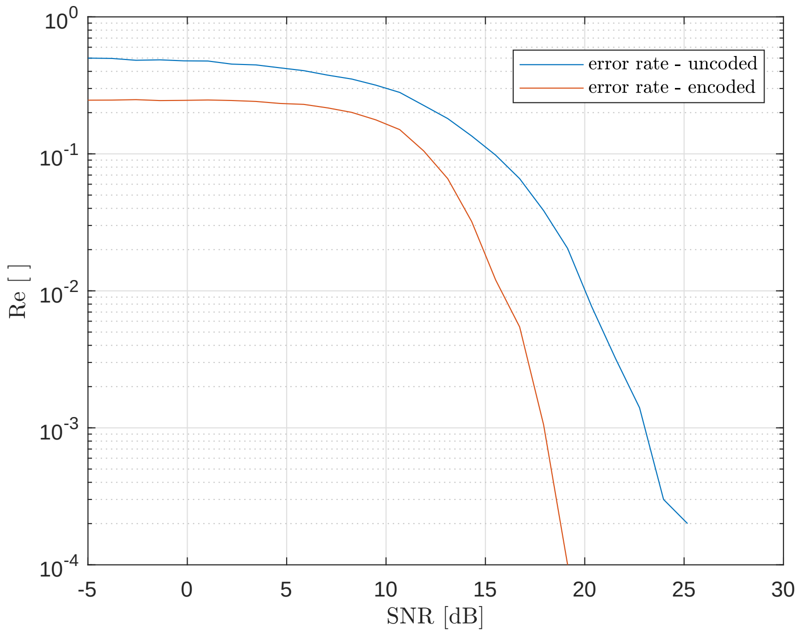


Figure 5.13: Real metric, encoded and uncode case, $h = 0.9\exp(j\pi/3)$ - QPSK

Chapter 6
Conclusion

The goal of this thesis was to get acquainted with Wireless Physical Layer Network coding. In the beginning, we introduced classical digital communication theory for point-to-point communication, including AWGN channel model. We also mentioned definition and properties of convolutional codes and its decoding. In the next chapter, we provided an introduction to WPNC coding. The necessary terms were defined. Then, we took a look at three-source-node network topology which has a lot of unexplored areas. We focused on designing a proper HNC map and we paid attention to the H-constellation at the relay node. On top of that, we tried to find regions where these maps could be applied and we came up with two approaches that could give us a rough estimate. In the last chapter, a few simulations were done to verify the theory and to calculate Bit Error Rate (BER).

My contribution is the implementation of convolutional encoder and decoder which were implemented in Matlab. I also simulated the error-rate in two-way relay channel with BPSK and QPSK sources. We also confirmed that using convolutional code decreases transmission error rate. The three-source-node network was studied and a couple of HNC maps were designed for a relay node. Usability verification depending on the channel parametrization was investigated.

Further work could focus on the three-source-node network topology and find algorithms that could simplify the design. All points of the assignment were met.

Bibliography

[1] Jan Sýkora, Alister Burr. *Wireless Physical Layer Network Coding* Cambridge University Press, 2018

[2] Jan Sýkora, *Digital Communications - lecture slides*, 2018

[3] Soung Chang Liew, Shenhli Zhang, Lu Lu. *Physical-Layer Network Coding: Tutorial, Survey and Beyond* The Chinese University of Hong Kong, 2012

[4] Anton A. Huurdeman. *The Worldwide History of Telecommunications* John Wiley & Sons, 2003.

[5] Mischa Schwartz. *HISTORY OF COMMUNICATIONS* IEEE 2008

Appendix A

List of Matlab files

A.1 Functions

- `BN_euclidean_metric_BPSK.m`
- `BN_real_metric_BPSK.m`
- `BN_euclidean_metric_QPSK.m`
- `BN_real_metric_QPSK.m`
- `conv_code.m`
- `generate_trellis.m`
- `decode.m`

A.2 Scripts

- `data.m`
- `BER_simulation_BPSK.m`
- `BER_simulation_QPSK.m`