Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Radioelectronics

Wireless Physical Layer Network Coding with Compute and Forward Relaying Strategy

Diploma Thesis

Author: Bc. Jan Hejtmánek
Supervisor: prof. Ing. Jan Sýkora, Csc.

Assignment

Abstract

Wireless Physical Layer Network Coding with Compute & Forward Relaying Strategy

At the time Wireless Physical Layer Network Coding (WPNC) is hot topic in research community due to properties allowing to increase the overall troughput in dense networks. The main goal of the thesis is to analyze Compute & Forward as one possible relaying strategy in WPNC. In order to work with this strategy WPNC fundamentals and basics of lattice based code designs with emphasis on its information theoretic implications are explained in the first part of the thesis. Second part of the thesis focuses on Compute & Forward relaying strategy as a special case of technique used for WPNC. Analysis starts with a basic application to a single stage network analyzing the performance (residual MSE of lattice misalignment, noise amplification) in relationship with lattice inflation scaling and modulo lattice combination coefficients. There is also performed comparison of different Hierarchical Network Code (HNC) maps. The main contribution of the thesis is design of Hierarchical Interference Cancellation (H-IFC) concept and its comparision with standard solution.

Keywords

Wireless Physical Layer Network, Compute and Forward, Hierarchical Interference Cancellation, Lattice codes.

Abstrakt

Wireless Physical Layer Network Coding s Compute & Forward strategií

V současné době patří Wireless Physical Layer Network Coding (WPNC) do středu zájmu výzkumníků v oblasti bezdrátových komunikací. Díky vhodnému zpracovnání informace dokážeme zvýšit propustnost i masivně interferující sítě. Cílem této práce je pojednání o Compute & Forward jako jedné z možných strategií na relayi. První část práce se dotýká základů WPNC a možnosti využití přirozené struktury Lattice kódů pro komunikaci v síti. V druhé části práce provádíme analýzu strategie Compute and Forward, zejména reziduální interference způsobenou nedokonalou ekvalizací na straně přijímače. Dále porovnáváme různé volby hierarchických map a jejich vliv na propustnost v síti. Hlavní příspěvek této práce je návrh Hierarchického potlačení interference a porovnání s klasickým přístupem.

Klíčová slova

Wireless Physical Layer Network, Compute and Forward, Hierarchické potlačení interference, Lattice kódy.

Acknowledgement

I would like to thank to prof. Ing. Jan Sýkora for his useful advices and comments. I would also like to thank to all my colleagues from the DiRaC group for their willingless to help within consultations.

The thesis was supported by the European Science Foundation through FP7-ICT DIWINE project, the Grant Agency of the Czech Technical University in Prague, Grant no. SGS14/081/OHK3/1T/13.

Proclamation

I declare that this diploma thesis was written independently and only references stated herein were used acording to "Methodical instruction about complying ethical principles during elaborating university theses".

In Prague on May 9^{th}, 2014 ....

Contents

List of Abbreviations

2-WRC Two Way Relay Channel

AF Amplify and Forward

AWGN Additive White Gaussian Noise

BC Broadcast Channel

CSE Channel State Estimation

CDMA Code Division Multiple Access

CF Compute and Forward

FDMA Frequency Division Multiple Access

H-CSE Hierarchical Channel State Estimation

H-IFC Hierarchical Interference Cancellation

H-SI Hierarchical Side Information

HDF Hierarchical Decode and Forward

HNC Hierarchical Network Code

IT Information Theory

IF Integer Forcing

JDF Joint Decode and Forward

LDLC Low Density Lattice Codes

LDPC Low Density Parity check codes

LNC Linear Network Coding

MA Multiple Access

MAC Multiple Access Channel

MLAN Modulo Lattice Additive Noise

MMSE Mimimum Mean Square Error

MSE Mean Square Error

MTMN Multi-Terminal and Multi-Node

NC Network Coding

NIT Network Information Theory

PDF Probability Density Function

RC Relay Channel

SD Space Divison

SC Superposition Coding

SNR Signal to Noise Ratio

TDMA Time Division Multiple Access

VNR Volume to Noise Ratio

WPNC Wireless Physical Layer Network Coding

List of Figures

List of Tables

Part I

Theoretical background

1 Introduction to WPNC

In wireless communications, where many users want to communicate with each other, wireless channel sharing is issue that we have to deal. From linearity of environment, where electromagnetic waves representing signals from users propagate, follows that at receiver antenna we observe superposition of signals from n users

$$u(t) = \sum_k s_k(t), \tag{1.1}$$

where $s_k(t)$ is signal from k-th user in time domain. Our goal is to separate information from each user by proper signal processing. Currently used multiuser communication systems exploit traditional physical layer principles and techniques. These methods are based on orthogonality principle. It means that signals from users have zero mutual energy. We use elegant mathematical description for this principle given by

$$R_{ij} = \langle s_i(t), s_j(t) \rangle = 0 \ for \ i \neq j, \tag{1.2}$$

where $\langle \cdot, \cdot \rangle$ is notation used for inner product. This property can be reached by separation of signals in allocated resource. The main idea is in resource division to subchannels. By resource nature we distinguish following Multiple Access (MA) techniques - Time Division Multiple Access (TDMA), Frequency Division Multiple Access (FDMA), Code Division Multiple Access (CDMA) or Space Divison (SD). However, orthogonal division is suboptimal method, because this approach doesn't use all capabilities of the channel. Information Theory (IT) gives us results that by proper strategy we can use channel by more effective way. In further text we use many terms from this theory so it is important to explain IT fundamentals in the next section. Understanding the section 1.1 is important for the rest of this work. The section 1.2 briefly describes fundamentals and principles used in WPNC and its importance in future wireless networks.

1.1 Fundamentals of Information Theory

In this section only basic principles are described. For more detailed reading about IT we refer to [2]. We define some important terms and then fundamental theorems.

We understand the information as a message bringing something new about random variable that we observe. An interesting question is how to determine amount of information contained in such message. It is obvious that observing of known deterministic variable (the set of elementary events contains certain event) gives us no information. On other hand when we observe well blended random variable (elementary events are equiprobable), we are given a huge amount of information. This is natural way to measure uncertainty of random variable's outcome and we use it every day without any knowledge about Information Theory (IT). IT introduces elegant mathematical tool for this measurement.

Definition 1 (Entropy)**.** *Let x be a random variable with Probability Density Function (PDF) $p(x)$. Then function*

$$\mathcal{H}[x] = E_x[-\log_2(p(x))] \tag{1.3}$$

*is called **entropy** and is measured in bits.*

Note that entropy expresses mean value of information measure given by $-\log_2(p(x))$. In the case of continuous distribution operator $E[\cdot]$ is replaced its continuous representation (integral) and in the case of discrete distribution operator is replaced by discrete representation (weighted sum) but definition 1 holds for both cases. In continuous domain the function 1.3 is called **differential entropy**. If we replace PDF $p(x)$ from equation 1.3 by joint PDF $p(x, y)$ and compute mean over x and y we get joint entropy.

Definition 2 (Joint Entropy)**.** *Let x, y be random variables with joint PDF $p(x, y)$. The function*

$$\mathcal{H}[x, y] = E_{x,y}[-\log_2(p(x, y))] \tag{1.4}$$

*is called **joint entropy** and is measured in bits.*

Now we will investigate impact of one random variable observation on uncertainty about the other variables. We define two types of conditioned entropies.

Definition 3 (Outcome Conditioned Entropy)**.** *Let x,y be random variables and ξ be outcome of random variable x. Let$p(y|x = \xi)$ be conditioned PDF of y after observing outcome ξ of random variable x. The **outcome conditioned entropy** is defined as*

$$\mathcal{H}[y|x = \xi] = E_{y,x=\xi}[-\log_2(p(y|x = \xi))] \tag{1.5}$$

If we compute mean of outcome conditioned entropy over all possible outcomes of random variable x we get conditional entropy.

Definition 4 (Conditional Entropy)**.** *Let $\mathcal{H}[y|x = \xi]$ be outcome conditional entropy. **Conditional entropy** is defined as*

$$\mathcal{H}[y|x] = E_{x=\xi}[\mathcal{H}[y|x = \xi]] = E_{x,y}[-\log_2 p(y|x)] \tag{1.6}$$

We can interpret the equation 1.6 as average uncertainty about y after observing x. It is obvious that in case of independence of random variables x and y conditional entropy is equal to entropy of the whole random variable y.

Armed with terms entropy and conditional entropy we can define another important function, which describes measure of independence of two random variables.

Definition 5 (Mutual Information)**.** *Let x,y be random variables with PDFs p(x) and p(y). The function*

$$\mathcal{I}(y; x) = \mathcal{H}[y] - \mathcal{H}[y|x] \tag{1.7}$$

*is called **mutual information** of random variables x and y.*

It is useful to realize interpretation of equation 1.7. In the beginning of the trial our uncertainty about y is $\mathcal{H}[y]$. After observing x our uncertainty about y is reduced by $\mathcal{H}[y|x]$ so $\mathcal{I}(y; x)$ expresses amount of information about y contained in x. Now we derive properties of mutual information. As we said above when x and y are independent then $\mathcal{H}[y|x] = \mathcal{H}[y]$ and therefore mutual information of x and y is equal to zero. If x and y are identical random variables then $\mathcal{H}[y|x] = \mathcal{H}[y|y] = 0$ and therefore mutual information of x and y is equal to entropy of y. Because conditioning reduces entropy mutual information is always nonnegative. In the next paragraph we define the most important term associated with transferring information over communication channel.

We usually understand the communication channel as a pipe with two sides. One side is used as input and on the other side we observe the output of this pipe. It is obvious that amount of information which can be reliably transfered over the communication channel is limited.

Definition 6 (Capacity of the channel)**.** *Let p(x) be PDF of the channel input. Let y be output of the communication channel.* ***Capacity*** *is defined as maximum mutual information of channel input and output over all possible input distributions. Hence,*

$$C = \max_{p(x)} \mathcal{I}(x; y), \tag{1.8}$$

where C is capacity of the channel.

The channel capacity acts as capability of channel to reliable transfer information. In the coding theory we attempt to construct such codes that achieve the capacity.

Definition 7 (Rate of code)**.** *Let n be the codeword length and cardinality of codebook $\mathcal{A}$ be M. The code with this properties we called code (M,n). The function*

$$R = \frac{\log_2 M}{n} \tag{1.9}$$

is called ***code rate*** *and is measured in bits per channel usage.*

Theorem 1 (Channel coding theorem)**.** *Let n be the codeword length, the for any rates $R < C$ there exists $(2^{nR}, n)$ code with probability of error $P_e \to 0$. For any $(2^{nR}, n)$ with probability of error $P_e \to 0$ must hold that its rate $R < C$.*

The first part of theorem says that we are able to construct code with zero probability of error. Therefore, this part of theorem is sometimes called achievability theorem. The second part of the theorem, called the converse theorem, only expresses that any code with rate greater than capacity of the channel must have nonzero probability of error. Note that theorem 1 claims that there exist some codes with required properties. However, it does not give us hint how to construct such codes.

In the previous paragraphs we stated fundamental definitions associated with point to point communication over the channel. However, as we mentioned at the start of this book we work with multiuser communication systems (networks). Therefore, we need to make a short excursion to Network Information Theory (NIT). In multiuser communication system we can find certain topologies. The topology when many users compete for one destination is called **Multiple Access Channel (MAC)**. We talk about **Broadcast Channel (BC)** when one user want to simultaneously send information to many destinations. Typical topology for multiuser communications is also **Relay Channel (RC)** when one user helps the other two users in their point to point communication. General configuration is usually called **Multi-Terminal and Multi-Node (MTMN)**. Due to evolving networks and node operational constraints (e.g. node cannot simultaneously be transmitter and receiver) we define so-called multi-stage network.

In point to point case we defined term of channel capacity. For multiuser case it is not possible due to wireless channel sharing. Therefore, we introduce term capacity region.

Definition 8 (Achievability)**.** *Let $\boldsymbol{R} = [R_1, R_2, \ldots, R_K]$ be a vector of user rates. Given rate vector $\boldsymbol{R}$ is* ***achievable****, if it can be achieved by construction of some particular coding strategy.*

Definition 9 (Capacity Region)**.** *Union of all user rate vectors that are achievable under all possible transmission strategies we call* ***capacity region****.*

The typical example of capacity region is depicted in figure 1.1 by red curve. Unfortunately, we are able to exactly determine the capacity region in only few cases. In almost cases its very difficult task with high complexity and usually intractable. Therefore, we use so-called inner and outer bounds of capacity region. It is analogy of one-dimensional case, where we use some vicinity of the value that we are not able to determine exactly. These bounds define the area in which the border of capacity

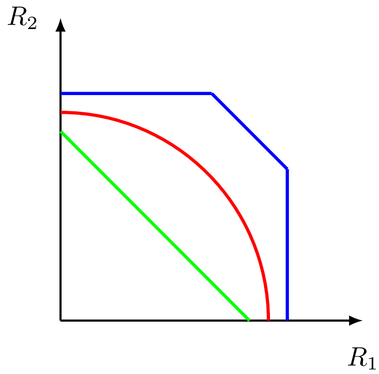


Figure 1.1: Capacity (red curve) and its inner(green curve) and outer (blue curve)

region lies. It is analogy of one-dimensional case, where we use some vicinity of the value that we are not able to determine exactly. The inner bound is always achievable, while the outer bound may not be achievable. The most popular one is called cut-set bound.

Theorem 2 (Cut-Set Bound)**.** *Assume that i-th node is equipped by $Tx = x_i$ and $Rx = y_i$. Let R_{ij} be rate from node i to node j. Assume independent uniformly distributed messages and node partition $(S, \bar{S})$. Then there exists a joint distribution $p(x_1, x_2, \ldots, x_k)$ such that any achievable rate R_{ij} is bounded by*

$$\sum_{i \in S, j \in \bar{S}} R_{ij} \leq \mathcal{I}(x^S; y^{\bar{S}} | x^{\bar{S}}) \tag{1.10}$$

for all $(S, \bar{S})$, where $x^S = \{x_i : i \in S\}, x^{\bar{S}} = \{x_j : j \in \bar{S}\}, y^{\bar{S}} = \{x_i : i \in \bar{S}\}$.

1.2 Wireless Physical Layer Network Coding

1.2.1 From network coding to Wireless network coding

WPNC is based on classical network coding principle used in ”wired” network [4]. In this paragraph we try to describe main idea, important for further understanding. Assume an arbitrary topology of nodes with arbitrary wired connection (or some isolated channel) between them. Each node can act as device forwarding all of its input information flow (so called relay). A traditional role of this relay is only repeating the received information flow (usually amplification). The main idea of Network Coding (NC) is in allowing any general relay operation instead of amplification. In typical scenario of NC relay receives packets from transmitting nodes and then sends only one or several packets which represent some combination of input information flow. This scenario and its benefit against traditional relay operation is closely described in the next example.

Example 1. *In figure 1.2 there is depicted an example of network, where node A and node B communicate each other. In classical approach communication is divided in four steps. At first node A sends the message to relay R. In the second step node B sends his message to relay. Then relay successively transmits stored messages to appropriate destination in two steps. NC approach can make this connection only in 3 steps. First two steps are the same, but in the third step relay transmits ”xor” operation of received messages. Then destinations are able to decode its desired information due to knowledge about own transmitted message in previous step.*

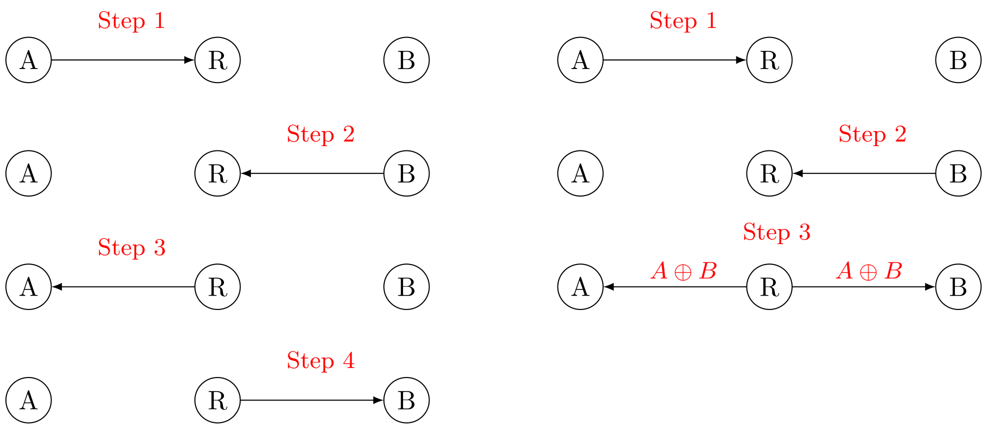


Figure 1.2: Number of steps needed to transfer information from node A to B and back

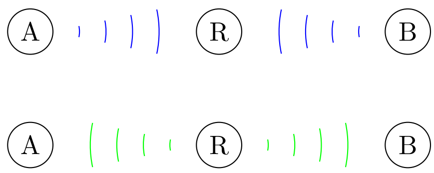


Figure 1.3: Number of steps reduction with WPNC approach. MAC stage is marked by blue color and BC stage by green.

From example above benefit of NC is obvious. Note that relay can combine input messages in general manner. Very popular relay operation is linear combination, which leads to so called Linear Network Coding (LNC).

Further, we show that we are able to save another step in topology described in example 1. From now, nodes communicate each other over wireless channel so we haven't luxury of isolated channels and relays observe only superposition of transmitted messages.

Example 2. *In Figure 1.3 there is shown simple network topology, where nodes communicate each other only trough wireless channel. Beside the example 1 the relay R observes superposition of messages from source nodes. Absence of isolated channels allows us to reduce number of steps needed to transfer information between nodes A and B, because one observation contains both messages simultaneously. On the other hand, it is obvious that relay operation can be more complex one.*

From principles mentioned above it follows that information flow in the network has hierarchical structure and it can be described by joint distribution of all involved subflows. This property of the flow is useful when we have partial knowledge about it. In previous examples this partial knowledge was represented by transmitted messages in last round. Because it helps to decode desired message from hierarchical information flow it is often called Hierarchical Side Information (H-SI). Interference in current wireless networks can be serious problem, but if we accept WPNC approach, the nuisance interference becomes to friendly one, because it brings partial information about the received flow structure. This is the main difference between current communication over wireless networks and WPNC.

The basic network entity for WPNC contains sources nodes, relay and destination nodes and is depicted in figure 1.4. Communication is divided into two stages. The first one, where many source nodes attempt to sent message to relay (MAC stage) and the second one, where relay distributes messages to many destination nodes simultaneously (BC stage). Hierarchical information flow processing can be handled by all

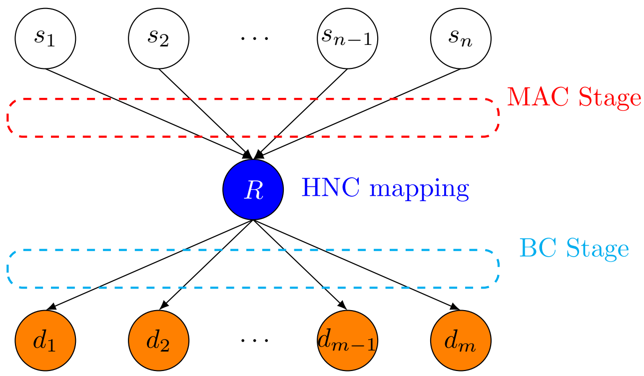


Figure 1.4: Basic network entity involving source nodes, relay and destination nodes.

nodes. For instance, the source node can generate the messages in such manner that it may be helpful for decoding it in receiver. So each node (source, relay, destination) can have specific job scenario, which helps to manage hierarchical information. In the next subsection relay nodes strategies are described.

1.2.2 Relaying strategies

The main goal of relay node is to create a hierarchical symbols from incoming messages. By comparison cardinalities of input and output alphabets we distinguish a several types of relay mappings (HNC - see definition 10).

Definition 10. *Let M_A and M_B be input alphabets cardinalities. Let M_{AB} be output alphabet cardinality. The HNC map is called*

- ***Full** if $M_{AB} = M_A \cdot M_B$*
- ***Minimal** if $M_{AB} = \min\{M_A, M_B\}$*
- ***Lossy** if $M_{AB} < \max\{M_A, M_B\}$*
- ***Extended** if $\max\{M_A, M_B\} < M_{AB} < M_A \cdot M_B$*

Now, we briefly describe what operation relay can use to create desired hierarchical structure.

Amplify and Forward (AF)

In this case relay only performs amplification of received symbols and then forwards it. Although this operation eliminates influence of spreading loss, the received symbol contains noise and it is also amplified. Increased noise power can be undesirable in some cases and then AF cannot be used by relay.

Joint Decode and Forward (JDF)

Relay decodes each input symbol individually and then hierarchical symbol is given by

$$\hat{d}_{A,B,\ldots} = \mathcal{X}(\hat{d}_A, \hat{d}_B, \ldots), \tag{1.11}$$

where $\mathcal{X}$ is HNC mapping. Thus, JDF produces noiseless symbols. However, during the decoding error can occur and as consequence of it hierarchical symbol can contain wrong data.

Hierarchical Decode and Forward (HDF)

Relay working under this strategy beside the JDF strategy performs estimate of HNC output. Hence the decoder and HNC mapper is one complex box and hierarchical symbol is given by

$$\hat{d}_{A,B,\dots} = \arg\max_{d_A,d_B,\dots} \mu\Big(\bigcup_{d_A,d_B,\dots:\mathcal{X}(d_A,d_b,\dots)=d_{AB}} \{d_A, d_B, \dots\}\Big), \tag{1.12}$$

where μ is proper decoding metric.

Compress and Forward

Relay applies the general nonlinear function on received information signal, usually quantization and then compression. The next relay scenario is a subset of this operation class.

CF

An interesting strategy by its natural approach. Main idea is in connection between linearity of environment and linearity of nested lattice codes. Because this text may mainly refer about CF, it is deeply explained in separate chapter. As we mentioned, the basic building block for CF approach is class of linear codes - Lattice codes. Therefore, the next chapter provides summary of lattice theory fundamentals.

2 Lattice codes

In previous chapter we discussed reasons for using WPNC concept in multi-user wireless networks. We also mentioned that CF as one of several relay strategies seems to be beneficial. This technique is based on lattice theory, which provides important results for WPNC applications. The goal of this chapter is to give basic definitions and results that will be extensively used in the sequel. In section 2.1 lattice codes are introduced as a special class of algebraic structures. Section 2.2 reveals the beauty of lattice codes - achieving the capacity of AWGN channel. Sections 2.4 describes extensions of lattice codes having practical potential use in CF.

2.1 Lattice as algebraic structure

Before starting to explain fundamental principles from lattice theory we need to state some basic definitions. In order to preserve continuity of this text all these important terms are defined at beginning of this section and in further we will refer to it.

Definition 11 (Lattice). *Lattice Λ is defined by*

$$\Lambda = \{\lambda = \boldsymbol{G}\boldsymbol{x} : \boldsymbol{x} \in \mathbb{Z}^n\}, \tag{2.1}$$

where $\boldsymbol{G}$ is a generating real-valued $n \times n$ matrix.

Definition 12 (Nested lattices). *Sequence of lattices $\Lambda, \Lambda_1, \Lambda_2, \ldots, \Lambda_L$ are nested if $\Lambda \subseteq \Lambda_1 \subseteq \Lambda_2 \subseteq \cdots \subseteq \Lambda_L$.*

Definition 13 (Coset). *Any translation of lattice, i.e. the set*

$$\Omega = \{\boldsymbol{x} + \Lambda : \boldsymbol{x} \in \mathbb{R}^n\} \tag{2.2}$$

is called ***coset*** *of Λ in $\mathbb{R}^n$.*

Definition 14 (Quantization). *Let $\boldsymbol{x} \in \mathbb{R}^n, \lambda \in \Lambda$. The function*

$$Q_\Lambda(\boldsymbol{x}) = \lambda \tag{2.3}$$

is called ***quantizer****. The error caused by quantization of $\boldsymbol{x}$ is given by $\boldsymbol{x} - Q_\Lambda(\boldsymbol{x})$.*

Definition 15 (Voronoi regions). *Let λ be an arbitrary lattice point. The set of points $\boldsymbol{x} \in \mathbb{R}^n$ fulfilling*

$$Q_\Lambda(\boldsymbol{x}) = \lambda \tag{2.4}$$

is called ***Voronoi region*** *of λ. Set of points $\boldsymbol{x} \in \mathbb{R}^n$ that after quantization give zero point is called the* ***fundamental Voronoi region*** *and is denoted by $\mathcal{V}_0$.*

Definition 16 (Covering Radius). *Let $\mathcal{B}(r)$ denotes an n-dimensional ball of radius r,*

$$\mathcal{B}(r) = \{\boldsymbol{s} : ||\boldsymbol{s}|| \leq r, \boldsymbol{s} \in \mathbb{R}^n\}. \tag{2.5}$$

Let $Vol(\mathcal{B}(r))$ denote its volume. The ***covering radius*** *r_{cov} of a lattice Λ is the smallest real number such that $\mathbb{R}^n \subseteq \Lambda + \mathcal{B}(r_{cov})$.*

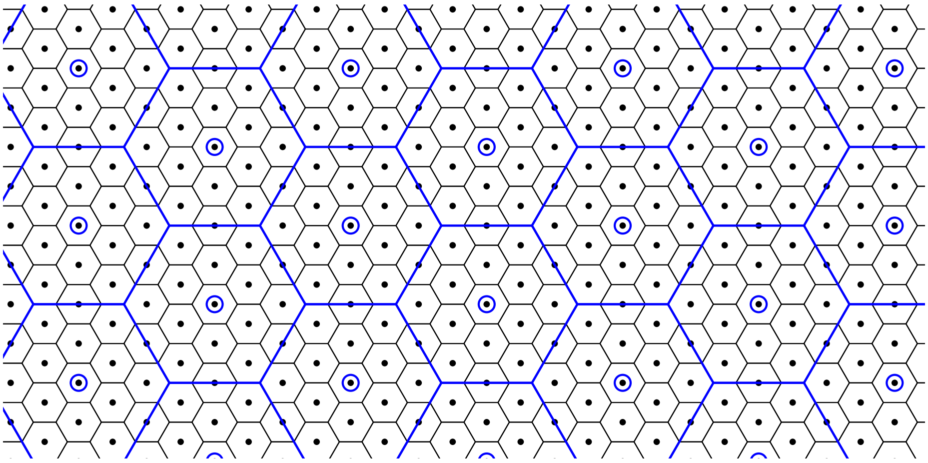


Figure 2.1: Example of nested lattices with appropriate Voronoi regions

Definition 17 (Effective radius)**.** *The* ***effective radius*** *of a lattice with Voronoi region $\mathcal{V}$ is the real number r_{effec} such that $Vol(\mathcal{B}(r_{effec})) = Vol(\mathcal{V})$.*

Definition 18 (Moments)**.** *The* ***second moment*** *of lattice Λ is defined as second moment per dimension of uniform distribution over the fundamental Voronoi region $\mathcal{V}$*

$$\sigma_\Lambda^2 = \frac{1}{n\,Vol(\mathcal{V})}\int_{\mathcal{V}} ||\boldsymbol{x}||^2 d\boldsymbol{x}. \tag{2.6}$$

The ***normalized second moment*** *is given by*

$$G(\Lambda) = \frac{\sigma_\Lambda^2}{(Vol(\mathcal{V}))^{\frac{2}{n}}}. \tag{2.7}$$

Definition 19 (AWGN goodness)**.** *Let $\boldsymbol{z} \in \mathbb{R}^n$ be an i.i.d Gaussian vector, $\boldsymbol{z} \sim \mathcal{N}(0, \sigma_Z^2 \boldsymbol{I}_{n\times n})$. Volume to Noise Ratio (VNR) is defined as*

$$VNR(\Lambda, \varepsilon) = \frac{(Vol(\mathcal{V}))^{\frac{2}{n}}}{\sigma_Z^2}, \tag{2.8}$$

where $\varepsilon = \Pr\{\boldsymbol{z} \notin \mathcal{V}\}$ induces σ_Z^2. We say that lattice Λ is good for AWGN if its VNR achieves $2\pi e$ for infinite dimensionality for each $\varepsilon \in (0, 1)$.

A lattice Λ is a discrete subgroup of Euclidean space with ordinary vector addition. This mean that if we take points $\lambda_1, \lambda_2 \in \Lambda$ then $\lambda_1 + \lambda_2$ also lies in Λ. From properties of group it also follows that each point $\lambda \in \Lambda$ has opposite point $-\lambda_1$ with respect to addition operator and this point is also element of lattice Λ. In (2.1) lattice is defined in terms of generating matrix $\mathbf{G}$. Thus, each lattice is created by taking all integer linear combinations of basis vectors (columns of $\mathbf{G}$). Example of lattice is depicted in figure 2.1, where Voronoi regions defined in def. 15 are also shown. It is useful to realize that a lattice together with appropriate Voronoi regions cover Euclidean space $\mathbb{R}^n$. Hence, each point $\mathbf{x} \in \mathbb{R}^n$ can be expressed as

$$\mathbf{x} = \lambda + \mathbf{r}, \tag{2.9}$$

where $\lambda \in \Lambda, \mathbf{r} \in \mathcal{V}$. Expression 2.9 can be transform to more generally form given by

$$\mathbb{R}^n = \Lambda + \mathcal{V}. \tag{2.10}$$

Equation 2.10 only describes $\mathbb{R}^n$ covering mentioned above. Vector $\mathbf{r}$ from 2.9 can be viewed as residual part of $\mathbf{x}$ after its quantization. Computation of $\mathbf{r}$ is usually called **modulo-Λ** operation denoted by

$$\mathbf{r} = \mathbf{x} \bmod \Lambda = \mathbf{x} - Q_\Lambda(\mathbf{x}). \tag{2.11}$$

For all $\mathbf{x}, \mathbf{s} \in \mathbb{R}^n, a \in \mathbb{Z}, \beta \in \mathbb{R}$ and $\Lambda \subseteq \Lambda_1$ the modulo-Λ operation satisfies following equalities:

$$\begin{aligned}
[\mathbf{x}+\mathbf{s}] \bmod \Lambda &= [\mathbf{s} \bmod \Lambda + \mathbf{x}] \bmod \Lambda && (2.12)\\
[Q_{\Lambda_1}(\mathbf{s})] \bmod \Lambda &= [Q_{\Lambda_1}(\mathbf{s} \bmod \Lambda)] \bmod \Lambda && (2.13)\\
[b\mathbf{s}] \bmod \Lambda &= [b[\mathbf{s} \bmod \Lambda]] \bmod \Lambda && (2.14)\\
\beta[\mathbf{s}] \bmod \Lambda &= [\beta\mathbf{s}] \bmod \Lambda && (2.15)
\end{aligned}$$

If Λ contains lattices distinct of itself than by definition 12 these sublattices are nested. Consider that we have two lattices Λ_1, Λ fulfilling $\Lambda \subseteq \Lambda_1$. In lattice theory Λ_1 is called fine, while the term of coarse lattice is used for Λ. This configuration is depicted in figure 2.1. Nested lattices can be used as so-called nested lattice codes, where the codebook $\mathcal{C}$ is the set of all points of fine lattice Λ_1 within the Voronoi region $\mathcal{V}$ of coarse lattice Λ. Thus,

$$\mathcal{C} = \Lambda_1 \cap \mathcal{V} \tag{2.16}$$

The rate of such code is given by

$$R = \frac{1}{n}\log_2|\mathcal{C}| = \frac{\mathrm{Vol}(\mathcal{V})}{\mathrm{Vol}(\mathcal{V}_1)} \tag{2.17}$$

In this paragraph there is shown how to determine quality of given lattice. In lattice theory is used the term of the covering goodness, which uses ratio of covering and effective radii defined in 16 and 17. Hence, the sequence of lattices $\Lambda^{(n)}$ is good for covering if holds

$$\lim_{n\to\infty}\frac{r_{\mathrm{cov}}}{r_{\mathrm{effec}}} = 1. \tag{2.18}$$

A lattice quality can be measured in sense of quantization goodness. A lattice Λ is good for quantization if its second normalized moment $G(\Lambda)$ achieves $(2\pi e)^{-1}$ for infinite dimensionality of lattice. In wireless communications channel is usually approximated by AWGN addition to useful message. Therefore, it is practical to define some metric describing impact of lattice use in that channel. Such metric is a core of definition 19. In [6] there was shown that sequences of such lattices exist. For nested lattice codes course lattice having good properties for covering, quantization and AWGN is desirable. It is sufficient that fine lattice is good for AWGN.

Lattices may be constructed by different ways. However, due to its illustrative nature we explain process of lattice construction described in [1] (Construction A). Note that sequences of lattices built by this construction are simultaneously good for covering, quantization and AWGN.

Consider that we have coarse lattice Λ of dimension n with second moment equal to P. Let $\mathbf{B} \in \mathbb{R}^n$ be the generating matrix of this lattice. Appropriate fine lattice is defined by using procedure explained below.

In the first step, we take an arbitrary linear code given over $\mathbb{F}_p^k$ represented by generating matrix $\mathbf{G} \in \mathbb{F}_p^{n\times k}$. Note that p must be prime. Thus, the codebook is the set

$$\mathcal{C} = \{\mathbf{c} = \mathbf{G}\mathbf{w} : \ \mathbf{w} \in \mathbb{F}_p^k\}, \tag{2.19}$$

where $\mathbf{w}$ is k-tuple given from $\mathbb{F}_p^k$ and all operation in this step are over this finite field. In the second step, codebook $\mathcal{C}$ is projected to real numbers by operation $g(.)$ and scaled down by a factor p. Then we place a copy of it on each integer vector. Thus, codebook $\tilde{\Lambda}$ over $\mathbb{R}^n$ is given by

$$\tilde{\Lambda} = p^{-1}g(\mathcal{C}) + \mathbb{Z}^n. \tag{2.20}$$

Note that $\tilde{\Lambda}$ compression of the codebook to n-dimensional cube $\langle 0,1)^n$ and placement of its copy only generates of lattice structure.

In the last step, the codebook $\tilde{\Lambda}$ is rotated by matrix $\mathbf{B}$. Now, we have available appropriate fine lattice λ_1 which is good for AWGN and it is consistent with the coarse lattice Λ. From previous text it follows that fine lattice is generated from the linear codebook $\mathcal{C}$ by using

$$\Lambda_1 = \mathbf{B}[p^{-1}g(\mathcal{C}) + \mathbb{Z}^n]. \tag{2.21}$$

2.2 Achieving the AWGN channel capacity with lattice codes

In text above we mentioned that lattice codes are perspective for using in AWGN. In this section we summarize results of lattice theory, mainly that this class of linear codes can achieve capacity of AWGN channel. More details you can find in [3]. Within observation of lattice properties it is useful to transform n-dimensional AWGN channel described by

$$\mathbf{y} = \mathbf{x} + \mathbf{z} \tag{2.22}$$

into Modulo Lattice Additive Noise (MLAN) channel [3]. Input of this channel is a fundamental Voronoi region $\mathcal{V}$ which is usually called **shaping lattice**. Then the transmitter is given by modulo-Λ operation

$$\mathbf{x} = [\mathbf{t} - \mathbf{d}] \bmod_{\mathcal{V}} \Lambda, \tag{2.23}$$

where $\mathbf{t}$ is information codeword and vector $\mathbf{d}$ is dither uniformly distributed in $\mathcal{V}$. It can be shown that if $\mathbf{d}$ is uniformly distributed in $\mathcal{V}$ then $[\mathbf{t}-\mathbf{d}] \bmod \Lambda$ is also uniformly distributed in fundamental Voronoi region. The receiver has available observation $\mathbf{y} = \mathbf{x}+\mathbf{z}$, where $\mathbf{z}$ has properties mentioned in definition 19. Upon the reception $\mathbf{y}$ is multiplied by scaling factor α, of which importance will be specified later, and dither $\mathbf{d}$ is added. The result is reduced by modulo-Λ operation, giving

$$\mathbf{y}' = [\alpha\mathbf{y} + \mathbf{d}] \bmod \Lambda. \tag{2.24}$$

By modification of 2.24 we obtain

$$\mathbf{y}' = [\mathbf{t} + \alpha\mathbf{z} + (1-\alpha)\mathbf{d}] \bmod \Lambda. \tag{2.25}$$

We can see that observation of undithered transmitted codeword has fluctuations given by component $\alpha\mathbf{z} + (1-\alpha)\mathbf{d}$. This convex combination of Gaussian vector and vector with uniform distribution is also known as effective noise and in further it will be denoted by $\mathbf{z}'$. The lattice is scaled so that the second moment of $\mathcal{V}$ is P_X. Thus,

$$\frac{1}{n}\int_{\mathcal{V}} ||x||^2 \; d\mathbf{x} = \sigma^2(\mathcal{V}) = P_X \tag{2.26}$$

A good point is how to choose scaling coefficient α. With a unscaled lattice correct decoding occurs when the noise falls within the Voronoi region. When power of noise is too large, this original Voronoi region is completely contained in noise sphere. Hence, the probability of correct decoding is proportional to the portion of noise sphere contained in original Voronoi region. With inflated lattice decoder the probability of correct decoding is also proportional to the portion of noise sphere contained in Voronoi region. However, in this case Voronoi region is taken with respect to inflated lattice. This region is centered around imaginary transmitted point of the inflated lattice. By inflating of the lattice we increase relative portion of mentioned spheres and therefore probability of correct decoding increases. However, increasing of scaling factor α cannot be boundless, because for large α both spheres have zero intersect. It is obvious that there exists certain value of α, for which the probability of correct decoding will be maximal. This value is known as the MMSE coefficient given by

$$\alpha = \frac{P_X}{P_X + P_N} = \frac{\mathrm{SNR}}{1 + \mathrm{SNR}} \tag{2.27}$$

minimizing the mean square error between received and transmitted messages. It can be shown that by using this scaling factor the normalized second moment of the effective noise $\mathbf{z}'$ is upperbounded by 2.28.

$$\frac{1}{n}E[||\mathbf{z}'||^2] \leq \frac{P_X P_N}{P_X + P_N} \tag{2.28}$$

In previous MLAN was defined and now we study its capacity. As we said if the input $\mathbf{T} \sim \mathcal{U}(\mathcal{V})$ then output of MLAN channel is also uniformly distributed in $\mathcal{V}$. We use this fact for evaluation of general form of information rate. It is given by

$$\begin{aligned} \frac{1}{n}\mathcal{I}(\mathbf{T};\mathbf{Y}') &= \frac{1}{n}\mathcal{H}[\mathbf{Y}'] - \frac{1}{n}\mathcal{H}[\mathbf{Y}'|\mathbf{T}] && (2.29)\\ &= \frac{1}{n}\log V - \frac{1}{n}\mathcal{H}[\mathbf{z}'] && (2.30)\\ &= \frac{1}{2}\log\frac{P_X}{G(\Lambda)} - \frac{1}{n}\mathcal{H}[\mathbf{z}'], && (2.31) \end{aligned}$$

where 2.31 follows from the definition of normalized second moment (def. 18) and from 2.26. Since the capacity of AWGN channel is given by

$$\frac{1}{2}\log(1+\mathrm{SNR}), \tag{2.32}$$

it follows from data processing inequality (see [2]) that

$$\frac{1}{n}\mathcal{I}(\mathbf{T},\mathbf{Y}') \leq \frac{1}{2}\log(1+\mathrm{SNR}). \tag{2.33}$$

The entropy of $\mathbf{z}'$ is upperbounded by entropy of white Gaussian vector with second moment determined by α_{MMSE} coefficient. Therefore, we can rewrite 2.31 as

$$\begin{aligned} \frac{1}{n}\mathcal{I}(\mathbf{T},\mathbf{Y}') &\geq \frac{1}{2}\log\frac{P_X}{G(\Lambda)} - \frac{1}{2}\log(2\pi e\frac{P_X P_N}{P_X+P_N}) && (2.34)\\ &= \frac{1}{2}\log(1+\mathrm{SNR}) - \frac{1}{2}\log(2\pi e G(\Lambda^{(n)})) && (2.35) \end{aligned}$$

We can see that component $\frac{1}{2}\log(2\pi e G(\Lambda^{(n)}))$ expresses distance from the capacity of the AWGN channel. This distance will be zero only for sequences of lattices, which are good for quantization. Thus, lattices, of which normalized second moment achieves $(2\pi e)^{-1}$.

2.3 Coding and decoding scheme for nested lattices

This section summarizes use of nested lattice codes in channel with additive white Gaussian noise. In previous section it was shown that nested lattice codes fulfilling certain considerations achieve AWGN channel capacity. Therefore, it is possible to replace traditional random codes in coding part of communication chain.

We will use such nested lattice codes that the coarse lattice Λ will be good for quantization, while the fine lattice Λ_1 will be a good channel code. We will try to incorporate these codes to MLAN channel described in previous section. Modulo lattice operation is over the Voronoi region of the coarse lattice Λ, while quantizer uses the Voronoi region of the fine lattice Λ_1. Assume that transmission and additive effective noise have the same nature as in derivation of MLAN channel. Hence, the received codeword will be obtained by

$$\begin{aligned} \hat{\mathbf{c}} &= Q_{\mathcal{V}_1}(\alpha\mathbf{y}+\mathbf{d}) \bmod \Lambda && (2.36)\\ &= Q_{\mathcal{V}_1}(\mathbf{c}+\mathbf{z}') \bmod \Lambda. && (2.37) \end{aligned}$$

From 2.37 it follows that the decoding error occurs when the effective noise vector $\mathbf{z}'$ is out of the fundamental Voronoi region of fine lattice Λ_1. Thus,

$$P_e = \Pr(\mathbf{z}' \notin \mathcal{V}_1). \tag{2.38}$$

It is important to realize that properties of effective noise determine the optimal decoding metric. By assumption of Gaussian nature the distance from the lattice point is the optimal. This is also known as Euclidean lattice decoding. However, as we saw in previous section the effective noise has generally different distribution. Therefore, Euclidean lattice decoding cannot be further optimal. In this case, the general Voronoi region is given by

$$\mathcal{V}^* = \{\mathbf{x} : p_{\mathbf{z}'}(\mathbf{x}) \geq p_{\mathbf{z}'}(\mathbf{x} - \mathbf{c} \bmod \Lambda),\ \forall \mathbf{c} \in \mathcal{C}\}, \tag{2.39}$$

where $p_{\mathbf{z}'}(.)$ denotes PDF of effective noise. A decoder using the quantizer $Q_{\mathcal{V}^*}(.)$ is usually called noise-matched lattice decoder. Achieving capacity with these decoders is point of the theorem 3.

Theorem 3. *For any $\varepsilon > 0$ there exists a sequence of n-dimensional nested lattice pairs $(\Lambda^{(n)}, \Lambda_1^{(n)})$ of which rate $R > C - \varepsilon$ for sufficiently large n and decoding error probability goes to zero with $n \to \infty$*

$$P_e = \Pr(\boldsymbol{z}' \notin V^*) \to 0. \tag{2.40}$$

It was shown that nested lattice codes can achieve the AWGN channel capacity. However, infinite dimension of lattice codeword makes this coding scheme practically unrealizable. The next section gives modification of nested lattice codes having potential in practical use.

2.4 Low Density Lattice Codes (LDLC)

We can see that lattice codes are Euclidean space analogy of linear block codes over finite field. Therefore, this analogy can be applied also on linear codes defined by its parity check matrix, so-called parity check codes. Encoding for the case of finite field and Euclidean space differs only in matrix and vector elements. Thus, encoder output equation (2.1) is the same for both cases. More interesting issue is Euclidean nature of syndrome decoding. In finite field syndrome is determined by

$$\mathbf{s} = \mathbf{H}\mathbf{x}, \tag{2.41}$$

where $\mathbf{x}$ is received codeword. From the fact that rows of $\mathbf{H}$ are basis vectors of null space of given code it is obvious that received codeword is valid one if and only if the appropriate syndrome has all elements equal to zero. Nonzero syndrome can be understood as projection of received codeword into error space. Because in the Euclidean case the parity check matrix $\mathbf{H}$ is defined as inverse matrix of $\mathbf{G}$ the result of 2.41 must have all integer elements. Otherwise, received codeword isn't valid one. Then the syndrome is associated with fractional part of given syndrome. Now we can see interesting property of decoder. Function of lattice syndrome decoder is nothing else than quantization. Hence, the syndrome is given by quantization error vector given by definition 14.

Equivalently as Low Density Parity check codes (LDPC) in finite field case, we can use its principle in lattice theory. In [7] there is shown that the lattice with generating matrix implicating sparse parity check matrix and with increasing dimension n can be getting closer to AWGN channel capacity. In following subsections we summarize the basic usage of such codes.

2.4.1 LDLC construction

The sparse matrix is such matrix that have only low number of nonzero elements with respect to its dimension. For purpose of construction of this matrix it is useful define row and column degrees.

Definition 20 (Row and Column degrees)**.** *Elements of the set* $\{\boldsymbol{r}_i\}_{i=1,\dots,n}$ *are called* ***row degrees*** *and the i-th element of this set determines the number of nonzero elements in the i-th row of* $\boldsymbol{H} \in \mathbb{R}^{n\times n}$*.*

Elements of the set $\{\boldsymbol{c}_i\}_{i=1,\dots,n}$ *are called* ***column degrees*** *and the i-th element of this set determines the number of nonzero elements in the i-th column of* $\boldsymbol{H} \in \mathbb{R}^{n\times n}$*.*

Definition 21 (Latin Square)**.** *LDLC is called* ***regular*** *if all rows and columns of matrix* $\boldsymbol{H}$ *have the same degree* $\boldsymbol{d}$*. LDLC is called* ***Latin square*** *if every row and column has the same* $\boldsymbol{d}$ *nonzero values, except of different sign.*

The parity check matrix $\mathbf{H}$ constitutes a bipartite graph of given code, where one set of nodes represents variable nodes (elements of codeword) and the second one incorporates all check nodes (parity check equations). Understanding the code structure as a graph is useful for decoding. In order to simplify decoder implementation it is useful to design matrix $\mathbf{H}$ is such way leading to appropriate loopless bipartite graph.

Now we focus on the Latin square LDLC. For construction such code is sufficient to determine generating sequence of nonzero elements in matrix $\mathbf{H}$ and required dimension n. All rows and columns are just permutations of each other. Permutations are placed in such manner that resulting graph is loopless. For encoding scheme described in following subsection is important to perform permutations in way leading to nonzero diagonal elements. The example of algorithm for construction parity check matrix with required properties is described in Appendix.

2.4.2 Encoding

We showed that we are able to construct desired sparse matrix $\mathbf{H}$. Unfortunately, appropriate generating matrix is not sparse in general and multiplication of integer vector by matrix $\mathbf{G}$. In order to simplify encoding operation using sparse parity check matrix is suggested [7]. Resulting codeword is then computed by iterative way. Iteration step is given by

$$\tilde{\mathbf{b}} = \mathbf{D}^{-1}\mathbf{b} \tag{2.42}$$

$$\mathbf{X} = \mathbf{D}^{-1}(\mathbf{U}+\mathbf{L}) \tag{2.43}$$

$$\mathbf{x}^{t+1} = \tilde{\mathbf{b}} - \mathbf{X}\mathbf{t}^{t}, \tag{2.44}$$

where matrices $\mathbf{D}$,$\mathbf{L}$,$\mathbf{U}$ are obtained by parity check matrix decomposition to diagonal, lower triangular and upper triangular matrix. Algorithm is initialized by zero vector $\mathbf{t}^0$ and finishes when relative distance between current and previous codeword is in desired tolerance limit. Implementation using MATLAB can be found in Appendix.

2.4.3 Lattice shaping

From definition of lattice it is obvious that codebook contains infinity number of codewords. In almost cases we work with power constrained AWGN channel where the whole lattice codebook is unusable. Therefore, some shaping method must be used to bound power of used codewords. For purposes of compute and forward relay operation nested lattice shaping, described in following paragraph, seems to be beneficial.

Nested lattice shaping method is based on use of pre-coder for integer vector $\mathbf{b}$ and result of this operation serves as the input for lattice encoder mentioned above. Assume that we wish to find such integer vector $\mathbf{k}$ minimizing

$$||\mathbf{b} - L\mathbf{k}||^2, \tag{2.45}$$

where $L \in \mathbb{R}$. When we are able to find optimal solution of this optimization task, we use it to pre-encode $\mathbf{b}$ to $\mathbf{b}'$ by

$$\mathbf{b}' = \mathbf{b} - L\mathbf{k}_{opt}. \tag{2.46}$$

If we apply the generating matrix to $\mathbf{b}'$, we obtain

$$\mathbf{G}\mathbf{b}' = \mathbf{G}\mathbf{b} - L\mathbf{G}\mathbf{k}_{opt}. \tag{2.47}$$

Now we can see interesting implication. Equation 2.47 is nothing else then quantization error resulting from modulo coarse lattice operation. Thus, the same principle as for general lattices can be applied to LDLC.

2.4.4 Decoding

In decoding procedure the main task is to find the nearest lattice point to our observation. Since this task has exponential complexity with respect to lattice dimension, similar iterative algorithm for decoding like LDPC must be used. It exploits bipartite graph form of parity check matrix, where one side of graph are variable nodes (lattice point elements) and the second one including check nodes (check equations). Connection between variable nodes and check nodes means that given variable nodes participate in one check equation together. Decoding procedure is described below.

Assume that we have available observation

$$\mathbf{y} = \mathbf{x} + \mathbf{z}, \tag{2.48}$$

where $\mathbf{x} \in \mathbb{R}^n$ is transmitted lattice point and $\mathbf{z} \sim \mathcal{N}(0, \sigma\mathbf{I})$ is additive white Gaussian noise. Let y_k denotes k-th element of the observation. Further we assume that we are able to construct bipartite graph with rules described above.

Initialization Each variable node x_m sends to all its check nodes the message

$$f_k^{(0)}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-y_k)^2}{2\sigma^2}}. \tag{2.49}$$

Check node message Each check node send different message to each of the variable nodes that are connected to it. Each check node represents appropriate check equation

$$\sum_{l=1}^{r} h_l x_{kl} = i, \tag{2.50}$$

where i is integer and r is row degree of matrix $\mathbf{H}$. The message that check node j sends back to variable node x_{kj} is calculated in following steps:

1. all received messages except message $f_j(x)$ are expanded by appropriate edge cost h_j
2. expanded PDFs are convolved

$$\tilde{p}_j(x) = f_1(\frac{x}{h_1}) * \cdots * f_{j-1}(\frac{x}{h_{j-1}}) * f_{j+1}(\frac{x}{h_{j+1}}) * \cdots * f_r(\frac{x}{h_r}) \tag{2.51}$$

3. message $\tilde{p}_j$ is then stretched by $-h_j$ to $\tilde{p}_j(x) = \tilde{p}_j(-h_j x)$
4. stretched result is extended to periodic function (because of unknown integer in check equation) with period $\frac{1}{|h_j|}$:

$$p_j(x) = \sum_{i=-\infty}^{\infty} \tilde{p}_j(x - \frac{i}{h_j}) \tag{2.52}$$

and result of this extension is then send to variable node x_{kj}.

Variable node message Each variable node sends different messages to all its check nodes that are connected to it. Assume that given variable node x_k is connected with check nodes denoted by $c_{k1}, c_{k2}, \dots, c_{ke}$, where e is appropriate column degree of $\mathbf{H}$. The message that variable node sends back to check node c_{kj} is calculated i following steps:

1. product step:

$$\tilde{f}_j = e^{-\frac{(x-y_k)^2}{2\sigma^2}} \prod_{l=1, l\neq j}^{e} p_j(x) \tag{2.53}$$

2. normalization step:

$$f_j(x) = \frac{\tilde{f}_j(x)}{\int_{-\infty}^{\infty} \tilde{f}_j(x)dx} \tag{2.54}$$

Final decision After reaching desired number of iterations we estimate the final PDFs of codeword elements x_k by calculating the variable node messages at the last iteration without omitting any check node message in the product step. Then we find peak in PDF for each variable node. The codeword $\hat{\mathbf{x}}$ composed from estimated elements is the nearest point to given observation $\mathbf{y}$. Then the integer vector is obtained from $\lfloor \mathbf{H}\hat{\mathbf{x}} \rceil$.

Derivation of this decoding algorithm and more details are derived in [7]. The following chapter discuss about CF relaying strategy in single stage scenario. It focuses on theoretical applications in WPNC.

3 Compute and Forward

Compute and Forward (CF) seems to be beneficial regard to its natural approach. In this chapter we focus on this relay operation and aspects arising from using it in WPNC. In the section 3.1 we show general scenario for CF using real and complex valued channels. The section also focuses on lattice inflation in order to reduce residual misalignment interference.In this section there is also shown how to optimally align channel coefficients to lattice structure.

3.1 Compute and Forward relaying strategy

3.1.1 Real valued channel

Consider general single stage network, where L sources wish to communicate with destination through M relays. Transmitter l wants to encode and send i.i.d. message $\mathbf{w}_l \in \mathbb{F}_p^{k_l}$. This message is processed by encoder, which is mapping $\varepsilon_l : \mathbb{F}_p^k \mapsto \mathbb{R}^n$ (k is maximal of k_l) and result is denoted $\mathbf{t}_l$. The codeword $\mathbf{t}_l$ is dithered and then modulo shaping lattice operation is performed. Thus, transmitted codeword is determined by

$$\mathbf{x}_l = [\mathbf{t}_l - \mathbf{d}_l] \bmod \Lambda. \tag{3.1}$$

Rate of this message is given by number of fine lattice points in fundamental Voronoi region of coarse lattice. It can be shown that we can determine the rate of message by

$$R_l = \frac{k_l}{n} \log_2 p, \tag{3.2}$$

where p is a prime. Since we assume linear AWGN channel each relay receives a noisy linear combination of transmitted codewords. The considered scenario is depicted in figure 3.1. Thus,

$$\mathbf{y}_m = \sum_{l=1}^{L} h_{ml}\mathbf{x}_l + \mathbf{z}_m, \tag{3.3}$$

where $h_{ml} \in \mathbb{R}$ are the channel coefficients and $\mathbf{z} \sim \mathcal{N}(0, \sigma^2\mathbf{I})$ is additive white Gaussian noise. Each of M relays attempts to recover desired equation of transmitted messages $\hat{\mathbf{v}}_m$ with integer valued coefficients. Thus, the equation is recovered as

$$\hat{\mathbf{v}}_m = [\sum_{l=1}^{L} a_{ml}\mathbf{x}_l] \bmod \Lambda. \tag{3.4}$$

If we compare expressions 3.3 and 3.4, we can see the main idea of CF strategy. Our task is to align received linear combination of transmitted codewords to lattice point in fundamental Voronoi region of shaping lattice Λ. Trivial case occurs when channel coefficients are integers. Hence, the relay observation is noisy lattice point and equation is obtained only by modulo-Λ operation. However, nonideal world makes our job quite more difficult, because channel coefficients are in almost cases real (resp. complex) valued. In order to deal with this problem, before modulo operation simple

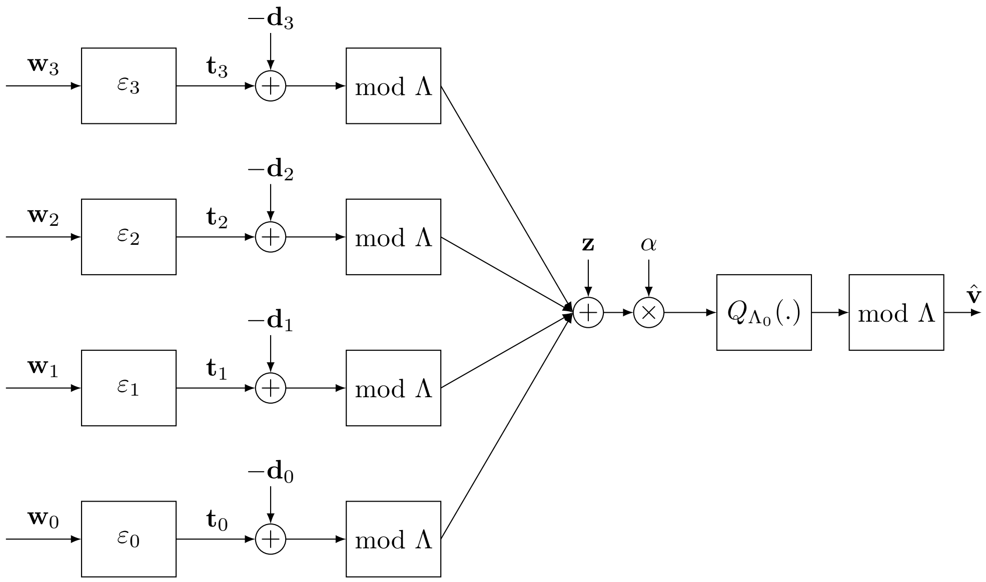


Figure 3.1: MAC channel with AWGN, where $L = 4$ sources wish to communicate with one relay.

one-tap equalization is performed. This equalization can be assigned with lattice inflation mentioned in previous chapter.

Because of noninteger channel coefficients, residual interference proportional to lattice misalignment appears among the noise components. Power of such interference is given by

$$P_{ri} = \mathbb{E}[||\alpha \sum_{i=1}^{L} h_{ml}\mathbf{x}_l - \sum_{l=1}^{L} a_{ml}\mathbf{x}_l||^2], \tag{3.5}$$

where $\alpha \in \mathbb{R}$ is an inflation coefficient. In order to minimize (3.5) for given vector $\mathbf{a}$ we attempt to find optimal α aligning channel coefficients to na lattice structure. Due to optimization task nature, optimal inflation coefficient is called Mimimum Mean Square Error (MMSE) coefficient and is obtained by

$$\alpha_{MMSE} = \frac{P\mathbf{h}_m^T\mathbf{a}_m}{N + P||\mathbf{h}_m||^2} \tag{3.6}$$

where $\mathbf{h}_m$ denotes vector of channel coefficients with respect to m-th relay and $\mathbf{a}_m$ is vector including coefficients of appropriate desired equation. The value of residual interference power influences maximal achievable rate for each user.

Definition 22 (Computation rate)**.** *Let $\boldsymbol{h} \in \mathbb{R}^L$ be a vector of channel coefficients that should be aligned to a given vector $\boldsymbol{a} \in \mathbb{Z}^L$. Let P denotes mean power of transmitted codewords. Then l-th user can use nested lattice code with rate r_l upperbounded by*

$$r_l < \frac{1}{2}\log_2^+ \frac{P}{\alpha_{opt}^2\sigma^2 + ||\alpha_{opt}\boldsymbol{h} - \boldsymbol{a}||^2 P} \tag{3.7}$$

*and relay R can reliably recover desired equation with probability $1 - \varepsilon$ for any $\varepsilon > 0$. Right hand side of inequality (3.7) is called **computation rate**.*

Computation rate is achievable by decoding process described above (inflation, removing dithers, quantization, modulo lattice). So far we discussed about equation recovering. However, we wish to decode all transmitted codewords, so we must recover sufficient number of independent equations with respect to a number of unknowns.

Such equations can be found by miscellaneous ways. The first one, so-called Integer Forcing (IF) exploits, appropriate set of linearly independent integer vectors with the highest computation rates [5]. Note that when each user use the same nested lattice pair, overall computation rate is given by the lowest of them. This difficulty can be solved by using nested lattice pairs with different rates. Then we are able to decode all codewords by smart successive decoding. The second approach is to use more independent observations (e.g., destination receives equations from sufficient number of relays). When we have available all equations, desired codewords are determined as solution of system of equations that can be solved over $\mathbb{R}^n$ or equivalently in $\mathbb{F}_p^n$. Finite field nature of desired equations is given by

$$\mathbf{u} = \bigoplus_{i=1}^{L} q_i \mathbf{w}_i, \tag{3.8}$$

where $q_i = g^{-1}([a_l]\text{mod}p)$.

3.1.2 Complex valued channel

In previous subsection we restricted the channel gains to be real valued. However, the most common case is that coefficients are complex numbers, because channel performs some attenuation and phase rotation. In this subsection we extend strategy mentioned above to complex numbers domain. The channel model is the same as in (3.3), but domains of participants are different. In complex valued channel we can exploit theory from real case described above. Sources transmit n-dimensional complex valued codewords including two real valued codewords (one codeword per its real and imaginary part). This is an analogy of traditional Superposition Coding (SC) technique. Thus, user l sends the codeword x_l, which is result of

$$\mathbf{x}_l = [\mathbf{t}_l^R - \mathbf{d}_l^R] \text{ mod } \Lambda + j[\mathbf{t}_l^I - \mathbf{d}_l^I] \text{ mod } \Lambda. \tag{3.9}$$

In this case, we consider the same scenario as for real valued channel gains. Therefore, relay has available observation in form

$$\mathbf{y}_m = \sum_{l=1}^{L} h_{ml} \mathbf{x}_l + \mathbf{z}_m, \tag{3.10}$$

where $hml \in \mathbb{C}, \mathbf{x} \in \{\mathbb{Z}^n + j\mathbb{Z}^n\}$. Relay then performs single tap equalization by $\alpha_{MMSE} \in \mathbb{C}$ coefficient and removes dithers. In the next step, the output of equalizer is split to real and imaginary part. We denote it by

$$\mathbf{s}_m^R = \Re[\alpha_m \mathbf{y}_m] + \sum_{l=1}^{L} \Re[a_{ml}]\mathbf{d}_l^R - \Im[a_m l]\mathbf{d}_l^I \tag{3.11}$$

$$\mathbf{s}_m^I = \Im[\alpha_m \mathbf{y}_m] + \sum_{l=1}^{L} \Im[a_{ml}]\mathbf{d}_l^R - \Re[a_m l]\mathbf{d}_l^I \tag{3.12}$$

Both streams are further processed in the same manner as in real valued case in order to get estimates of desired lattice equations

$$\hat{\mathbf{v}}_m^R = [Q_{\Lambda_m}(\mathbf{s}_m^R)] \text{ mod } \Lambda, \tag{3.13}$$

$$\hat{\mathbf{v}}_m^I = [Q_{\Lambda_m}(\mathbf{s}_m^I)] \text{ mod } \Lambda. \tag{3.14}$$

Processing of real and imaginary part of the result is identical with real valued case. It is important to realize that in one complex lattice point we send double information amount in comparison with real valued case. Thus, the resulting computation rate is given by

$$R_{cmp} = \log_2^+ \frac{P}{\alpha_{opt}^2 \sigma^2 + ||\alpha_{opt}\mathbf{h} - \mathbf{a}||^2 P}. \tag{3.15}$$

Part II

Contribution of the thesis

4 Selection of HNC maps for Compute and Forward

4.1 Residual lattice misalignment analysis

In section 3.1 we determined upper bound of source rates for given **a**. From (3.7) it can be seen that inappropriately chosen integer coefficients can cause significant degradation of achievable transmission rate. In this section we suggest possible methods to design proper integer vector with desired performance. Note that choice of optimal HNC map (desired equation) for given **h** is momentally ignored issue by research community. In further text we assume complex scenario described in subsection 3.1.2 and 3-user MAC channel. Therefore, computation rate is given by 3.15. We now introduce three intuitive ways:

Channel coefficients rounding Intuitive quantization of channel coefficient leads to solution, which is relatively close to channel. Below, we show that this operation can provide suboptimal HNC.

Rationalization We can express channel coefficients as a rational number. And after proper scaling we obtain optimal solution leading to zero lattice misalignment. However, in almost cases, the resulting HNC map is too far from channel coefficients. This fact causes that optimal α can achieve large values implying undesirable noise amplification. Therefore, this method is optimal only for high SNR levels.

Brute force Possible solution is to generate integer grid in sufficiently large vicinity of channel coefficients. Then we compute maximal achievable rate for each element of grid and find solution maximizing computation rate for given **h**. Note that this approach has increasing computational complexity with number of users.

In the next subsection, we investigate performance of these methods in real and complex valued channels.

4.1.1 Designed HNC map performance in real valued channel

Now, we analyze methods suggested above in two types of channel coefficients. In the first case, we selected channel vector **h** causing relatively small lattice misalignment. Without loss of generality we assume channel vector $\mathbf{h} = [2.9, 2.1, 0.9]$. It can be expected that HNC map given by rounding operation should lead to good performance. Computation rates for each method are shown in figure 4.1. Indeed, we can see that rounding strategy has relatively high computation rate for almost levels of SNR. One could argue that for large values of SNR the best choice is rationalized HNC map. However, rationalization can achieve zero residual lattice misalignment, but we pay for it by large value of α_{opt} (see figure 4.2). Therefore, rationalization

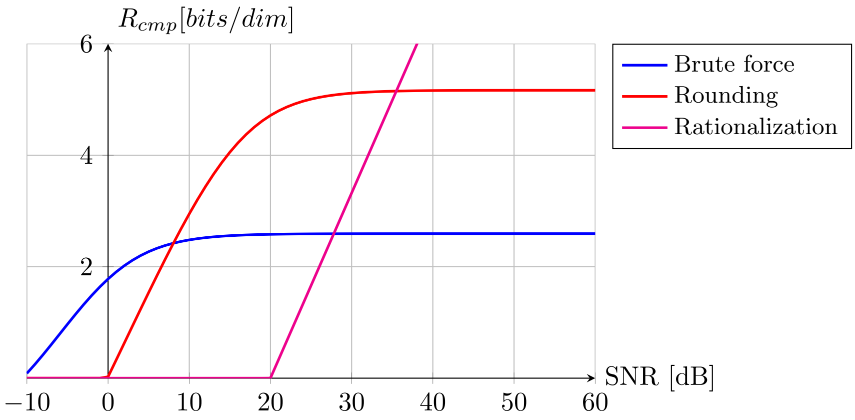


Figure 4.1: Computation rate for channel causing small lattice misalignment ($\mathbf{h} = [2.9, 2.1, 0.9]$)

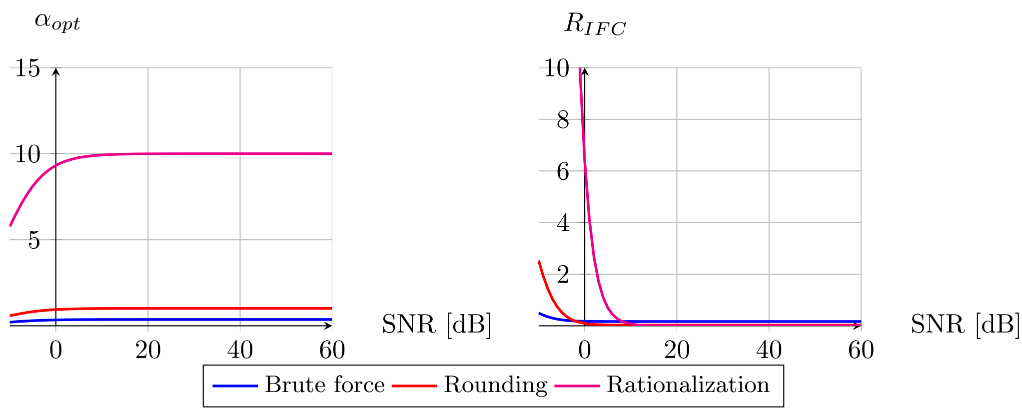


Figure 4.2: Channel with small lattice misalignment ($\mathbf{h} = [2.9, 2.1, 0.9]$). Optimal inflation coefficient for different HNC maps (left plot). Residual interference for different HNC maps (right plot)

is practically unusable in systems with low level of SNR. Brute force solution maximizing computation rate at SNR = 5 dB is optimal only for low SNRs. Hence, our expectation about optimality of rounding strategy is verified.

Now, assume that channel coefficients cause large lattice misalignment. Thus, consider channel vector $\mathbf{h} = [3.4, 2.3, 1.6]$. Computation rates for three different HNC maps are shown in figure 4.3. In this case, rounding causes significant degradation of computation rate, while brute force HNC map, selected as the best at SNR = 5 dB, seems to be optimal for almost SNR levels. For large values of SNR HNC map created by channel vector rationalization has the best performance. Residual interference caused by lattice misalignment after channel inflation converges to zero, but it is unusable at low levels of SNR because of too large noise amplification (see figure 4.4).

Numerical results show that for small lattice misalignment we can use desired equation with coefficients given by rounding operation applied to channel. However, we can see from figure 4.3 that such choice can lead to degradation of achievable source rates. In this case, it is important to properly choose map ensuring high computation rate for given SNR.

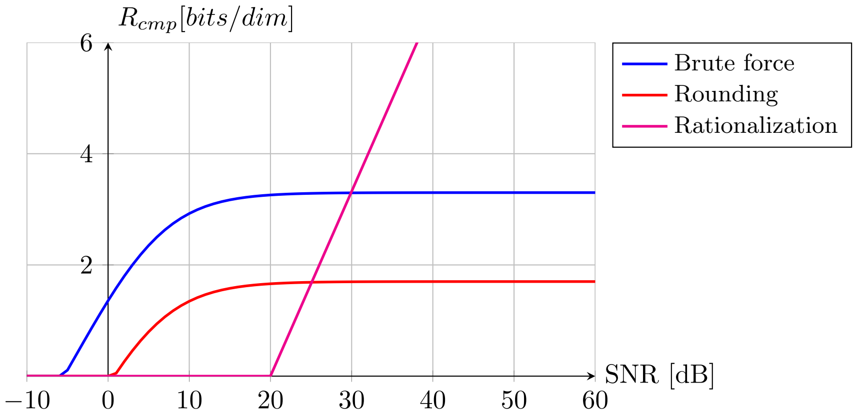


Figure 4.3: Computation rate for channel causing small lattice misalignment ($\mathbf{h} = [3.4, 2.3, 1.6]$)

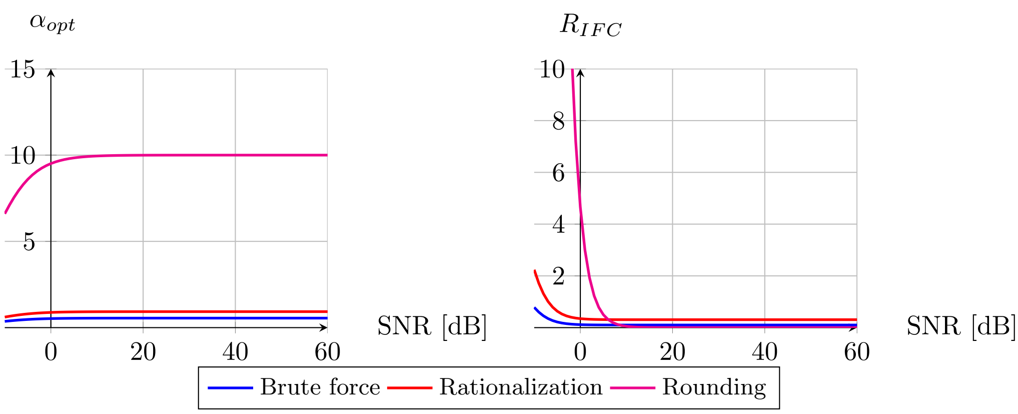


Figure 4.4: Channel with large lattice misalignment ($\mathbf{h} = [3.4, 2.3, 1.6]$). Optimal inflation coefficient for different HNC maps (left plot). Residual interference for different HNC maps (right plot).

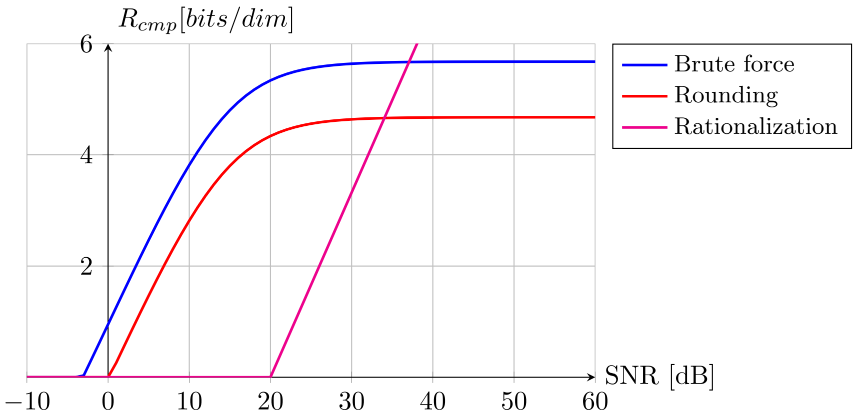


Figure 4.5: Computation rate for channel causing small lattice misalignment ($\mathbf{h} = [3.1 + 0.8j, 1.9 + 1.9j, 1.1 + 2.9j]$)

4.1.2 Designed HNC map performance in complex valued channel

In previous subsection we assumed only real valued channel coefficients. Now, we investigate behaviour of CF in complex valued channel. Again, we assume three types of HNC map created by operations described above. We begin with analysis of well behaving MAC channel described by vector $\mathbf{h} = [3.1 + 0.8j, 1.9 + 1.9j, 1.1 + 2.9j]$. Results drawn in figure 4.5 show that rounding strategy is not optimal for choice of HNC map, even for lattice coefficients similar to integers. In order to maximize computation rate at SNR=5 dB we selected $\mathbf{a}_{opt} = [2 - j, 2, 2 + j]$ by brute force. Interesting issue is that such map does not have intuitive form, but together with α_{MMSE} coefficient can achieve good results. Dependence of inflation coefficient on value of SNR for given map is depicted in figure 4.6. Again, for large SNR values we can choose rationalized channel gains as desired map giving zero residual interference. However, in the region of low SNR this map has disastrous performance.

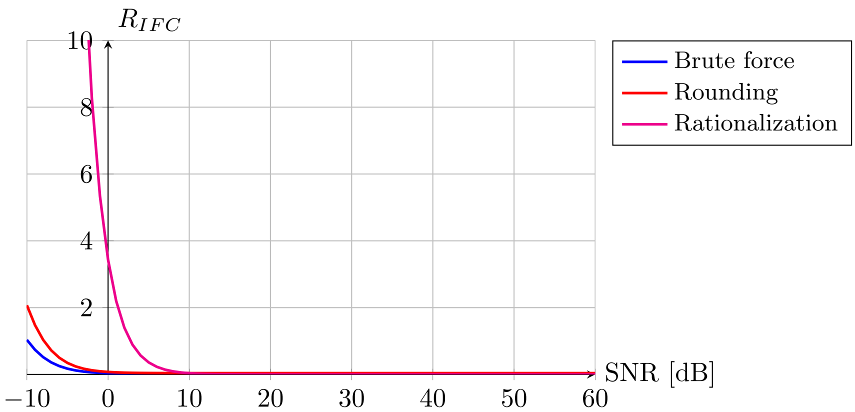


Figure 4.6: Channel with small lattice misalignment ($\mathbf{h} = [3.1 + 0.8j, 1.9 + 1.9j, 1.1 + 2.9j]$). Residual interference for different HNC maps.

We obtained similair results for case of large lattice misalignment, where optimal map had to be determined by brute force method.

We showed that we must properly choose HNC map, which influences achievable value of source rates. From our results it can be seen the computation rate as optimality criterion depends strongly on SNR level. In the case, when whole system works with variable noise power, we must choose such optimality criteria ensuring good results for almost SNR levels.

5 Hierarchical Interference Cancellation

We saw that HNC map selection essential influence to overall performance of the network. Inappropriately chosen map can cause too large residual interference. Therefore, resulting computation rate can be zero for all range of SNR values. We showed that always we are able to find HNC map with good results. More complicated is the case, when we want to perform H-IFC. The concept of H-IFC describes the technique where we help the decoder of given desired HNC map by partially mitigating the ambiguity generated by all involved sources by using other (again many-to-one) HNC map structure knowledge. Number of known maps is assumed not to be sufficient for fully resolving all contributing sources. This chapter provides techniques how to solve this issue with using the CF strategy.

5.1 Classical multimap CF

It might be happen that desired map does not have sufficient computation rate. In order to improve it we can decode multiple auxiliary HNC maps with good computation rate. These maps must give our desired map as their linear integer combination. Then desired map can be obtained by

$$\mathbf{a} = \sum_m \eta_m \tilde{\mathbf{a}}^m, \tag{5.1}$$

where $\eta \in \mathbb{Z}$ and $\tilde{\mathbf{a}}^m$ denotes m-th auxiliary map. Note that resulting computation rate is given by minimum between all HNC maps participating in 5.1. In [1] it is called as *successive CF*. We will use this concept as a reference case for comparison with our hierarchical interference cancellers.

5.2 Joint Hierarchical Interference Canceller

It might be happen that we are not able to find sufficient number of auxiliary maps giving desired map as linear integer combination. Thus, another way to decode desired map must be used. We suggest two concepts. The first one, described in this section, is called *Joint Hierarchical Interference Canceller* [8]. It is generalization of classical CF. Instead of use only single tap equalizer, we allow to decode desired map with auxiliary maps. Then we can use multi-tap scaling of auxiliary maps in order to minimize residual lattice misalignment.

Now we desribe Joint H-IFC in details. Let the desired map be $\mathbf{a}$ (it does not need to be a linear combination of auxiliary maps). Joint H-IFC multi-tap equalizer minimizes MSE lattice misalignment w.r.t. to desired map $\mathbf{a}$ including multi-tap scaling of all available decodable hierarchical codeword structures

$$[\hat{\alpha}, \hat{\boldsymbol{\beta}}] = \arg\min_{\alpha,\boldsymbol{\beta}} \mathrm{E}\left[||\alpha \mathbf{x} - \sum_m \beta_m \left(\sum_i \tilde{a}_i^m \mathbf{c}_i \right) - \sum_i a_i \mathbf{c}_i ||^2 \right], \tag{5.2}$$

where $\boldsymbol{\beta} = [\ldots, \beta_m, \ldots]$.

By using auxiliary maps we obtain more degrees of freedom to reduce residual lattice misalignment. From (5.2) it is clear, why we use the word *hierarchical.* In classical interference cancellation we successively subtract decoded codeword from observation. Instead, in our approach we subtract hierarchical functions (many-to-one) of involved codewords.

Now we investigate how to obtain scaling coefficients in closed form. In order to simplify the solution, we assume only single auxiliary map $\tilde{\mathbf{a}}$. Single auxiliary map Joint H-IFC MMSE equalizer is given by

$$[\hat{\alpha}, \hat{\beta}] = \arg\min_{\alpha,\beta} \mathrm{E}\left[||\alpha\mathbf{x} - \beta\sum_i \tilde{a}_i\mathbf{c}_i - \sum_i a_i\mathbf{c}_i||^2\right] \tag{5.3}$$

Finding $\hat{\alpha}, \hat{\beta}$ is optimization task, where utility function is

$$\begin{aligned} \rho &= \mathrm{E}\left[||\alpha\mathbf{x} - \beta\sum_i \tilde{a}_i\mathbf{c}_i - \sum_i a_i\mathbf{c}_i||^2\right] & (5.4)\\ &= nP||\alpha\mathbf{h} - \beta\tilde{\mathbf{a}} - \mathbf{a}||^2 + n|\alpha|^2\sigma^2. & (5.5)\end{aligned}$$

The function ρ is real valued of complex variables α and β. Therefore, in order to find stationary point we need to use generalized derivative. By using rules for generalized derivation we obtain closed form for optimal MMSE coefficients as a solution of

$$\begin{bmatrix} \gamma||\mathbf{h}||^2 + 1 & -\gamma\mathbf{h}^H\tilde{\mathbf{a}} \\ \gamma\tilde{\mathbf{a}}^H\mathbf{h} & -\gamma||\tilde{\mathbf{a}}||^2 \end{bmatrix} \begin{bmatrix} \hat{\alpha} \\ \hat{\beta} \end{bmatrix} = \begin{bmatrix} \gamma\mathbf{h}^H\mathbf{a} \\ \gamma\tilde{\mathbf{a}}^H\mathbf{a} \end{bmatrix}, \tag{5.6}$$

where $\gamma = \frac{P}{\sigma^2}$ denotes SNR. Computation rate after decoding auxiliary maps is then given by

$$R_j = \log_2^+ \frac{P}{|\alpha|^2\sigma^2 + P||\hat{\alpha}\mathbf{h} - \hat{\beta}\tilde{\mathbf{a}} - \mathbf{a}||^2}. \tag{5.7}$$

Numerical results and comparison with standard multiple map CF are shown in section 5.4.

5.3 Recursive Hierarchical Interference Cancellation

Hierarchical interference cancellation can be also performed by recursive way. We can use auxiliary maps with good performance to create effective channel causing smaller lattice misalignment with respect to desired map.

We now describe basic iteration of recursive H-IFC. Assume we wish to decode desired HNC map $\mathbf{a}$ and we have available maps with acceptable computation rate. Main task is to subtract the properly scaled m-th auxiliary map from the observation in order to obtain new output of the MAC channel given by

$$\mathbf{y}_m = \mathbf{y}_{m-1} - \hat{\beta}_m \sum_i \tilde{a}_i^m \mathbf{c}_i. \tag{5.8}$$

Coefficient $\hat{\beta} \in \mathbb{C}$ is chosen in order to minimize

$$\hat{\beta}_m = \arg\min_{\beta} \mathrm{E}\left[||\mathbf{y}_{m-1} - \hat{\beta}_m\sum_i \tilde{a}_i^m\mathbf{c}_i - \sum_i a_i\mathbf{c}_i||^2\right]. \tag{5.9}$$

We can also find closed form solution for $\hat{\beta}$. For simplicity, we derive the solution for single auxiliary map $\tilde{\mathbf{a}}$. The utility function has the form

$$\begin{aligned} \rho &= E\left[||\mathbf{y}_{m-1} - \hat{\beta}_m\sum_i \tilde{a}_i^m\mathbf{c}_i - \sum_i a_i\mathbf{c}_i||^2\right] & (5.10)\\ &= nP||\mathbf{h} - \beta\tilde{\mathbf{a}} - \mathbf{a}||^2. & (5.11)\end{aligned}$$

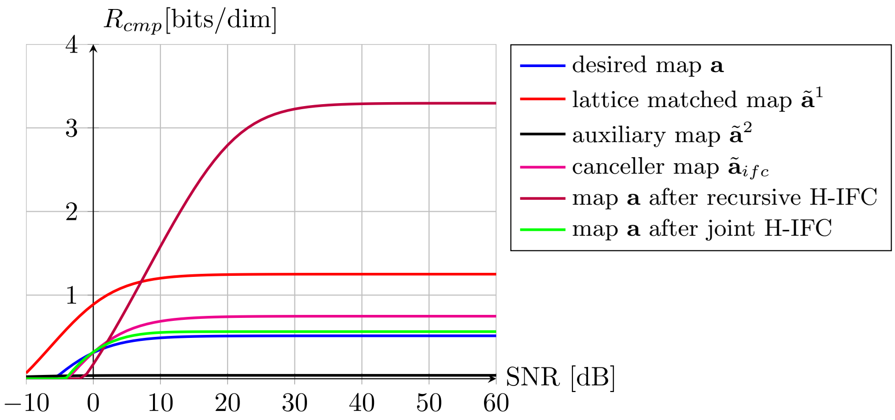
By performing generalized derivative in order to find stationary point of (5.11) we obtain interesting result. Optimal scaling coefficient has the form of the projector. Thus,

$$\hat{\beta} = \frac{\tilde{\mathbf{a}}^H(\mathbf{h} - \mathbf{a})}{||\tilde{\mathbf{a}}||^2}. \tag{5.12}$$

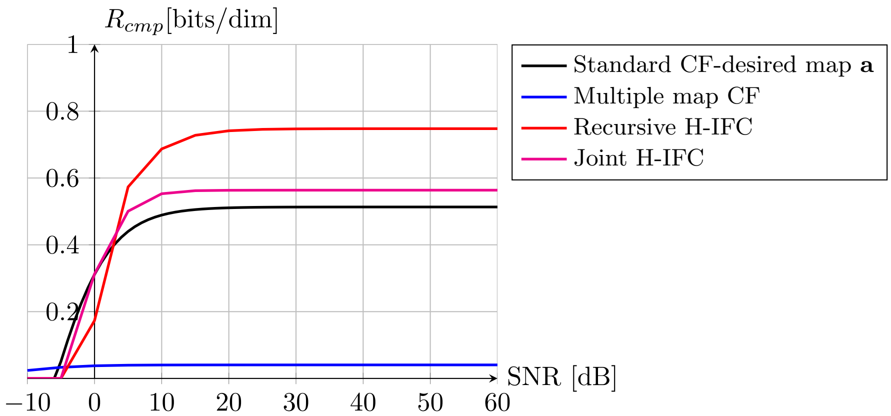
We can see that optimal scaling coefficient does not depend on SNR. The new effective observation together with another auxiliary map are then inputs for the next iteration. After performing the desired number of iterations, standard CF approach is then applied to resulting effective observation.

5.4 Numerical results

We examined both suggested one-step variants of hierarchical interference canceller in the 3-user MAC. Standard CF with multiple HNC maps was used as a reference scenario. In the first case, we considered channel coefficients causing only small lattice misalignment. Thus, we chose real valued vector of channel coefficients $\mathbf{h} = [3.1, 2.1, 0.9]$. Without loss of generality we assumed desired HNC map to be $\mathbf{a} = [1, 1, 1]$. In the reference case, we chose HNC map $\tilde{\mathbf{a}}^1 = \mathbf{a}_h = [1, 1, 0]$ maximizing the computation rate at SNR= 5 dB and second auxiliary map $\tilde{\mathbf{a}}^2 = \tilde{\mathbf{a}}_x = [0, 0, -1]$. These maps were selected in order to obtain the desired map $\mathbf{a}$ as their integer linear combination. For recursive and joint H-IFC auxiliary HNC maps, the map maximizing the computation rate of the desired map was used. For a given $\mathbf{h}$ we obtained $\tilde{\mathbf{a}} = \mathbf{a}_{ifc} = [-2, -1, 0]$. In figure 5.1a, there are plotted computation rates of individual maps. It is obvious that the resulting achievable rate of appropriate scenario is given by the minimal rate among all participating HNC maps. Achievable rates for all considered scenarios are plotted in figure 5.1b, where we can see that standard CF with multiple maps is not optimal for given $\mathbf{h}$, while both hierarchical interference cancelers gave the computation rates above the case of only one desired HNC map. The same analysis was performed for channel coefficients causing large lattice misalignment. An example of such channel is $\mathbf{h} = [3.3, 1.5, 0.9]$. Appropriate HNC maps together with their computation rates are shown in figure 5.2a. From results in figure 5.2 it can be seen that standard CF with chosen multiple maps has poor performance again. Recursive H-IFC gave better results, even without restriction on desired map to be a linear integer combination of auxiliary HNC maps. In [5] there is mentioned that CF extension to complex valued channel is straightforward, but all papers about CF give numerical results only for a real valued case. Therefore, we also show the results of considered scenarios for complex channel coefficients (figure 5.3a and figure 5.3b). Again we can see that recursive H-SD-IFC ranks among the scenarios with better performance. In figure 5.3b there is shown that CF with multiple auxiliary maps can also achieve good results, even better than joint H-IFC.

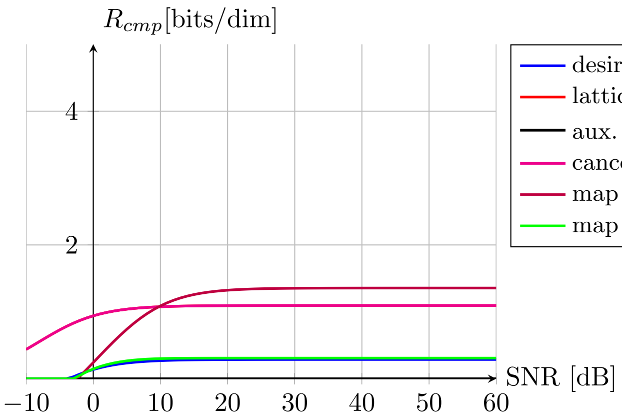


(a)

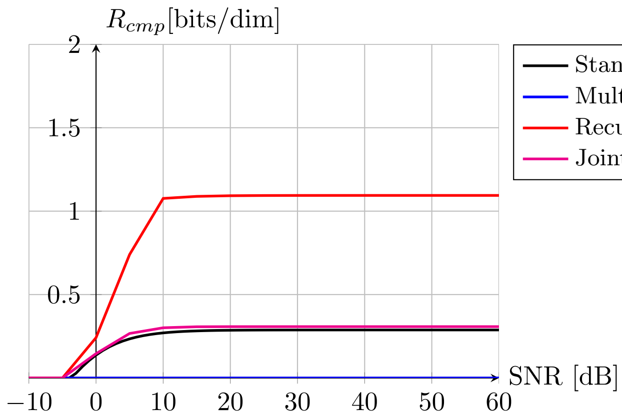


(b)

Figure 5.1: Channel $\mathbf{h} = [3.1, 2.1, 0.9]$, Desired map $\mathbf{a} = [1, 1, 1]$, aux. map $\tilde{\mathbf{a}}^1 = [1, 1, 0]$, aux. map $\tilde{\mathbf{a}}^2 = [0, 0, -1]$, H-IFC map $\tilde{\mathbf{a}}_{ifc} = [-2, -1, 0]$ $\rightarrow$ (a) Computation rates of individual HNC maps. (b) Resulting computation rate for given scenario as minimum over all participating maps.

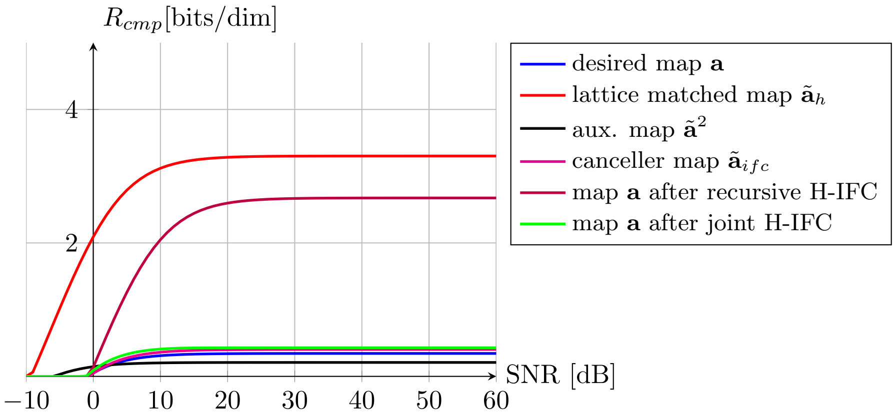


(a)

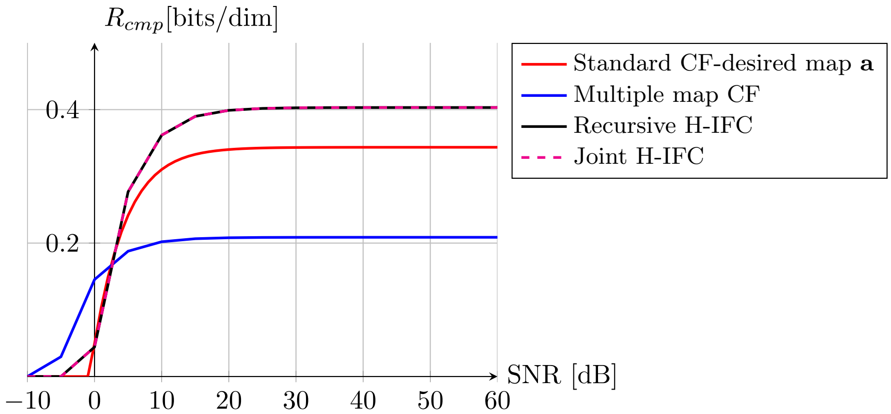


(b)

Figure 5.2: Channel $\mathbf{h} = [3.3, 1.5, 0.9]$, Desired map $\mathbf{a} = [1, 1, 1]$, aux. map $\tilde{\mathbf{a}}^1 = [1, 0, 0]$, aux. map $\tilde{\mathbf{a}}^2 = [0, -1, -1]$, H-IFC map $\tilde{\mathbf{a}}_{ifc} = [1, 0, 0]$ $\rightarrow$ (a) Computation rates of individual HNC maps. (b) Resulting computation rate for given scenarion as minimum over all participating maps.



(a)



(b)

Figure 5.3: Channel $\mathbf{h} = [3.4 + 0.4j, 2.3 + 1.7j, 1.6 + 2.4j]$, Desired map $\mathbf{a} = [1 + j, 1 + j, 1 + j]$, aux. map $\tilde{\mathbf{a}}^1 = [1 + j, j, j]$, aux. map $\tilde{\mathbf{a}}^2 = [0, -1, -1]$, H-IFC map $\tilde{\mathbf{a}}_{ifc} = [-2, -1 - j, -j]$ $\rightarrow$ (a) Computation rates of individual HNC maps. (b) Resulting computation rate for given scenarion as minimum over all participating maps.

6 Evaluation of results

In this chapter we comment all used scripts giving numerical results for previous text. Each important part of script is in details described in one paragraph. All results were evaluated with MATHEMATICA language.

6.1 HNC map optimality

In this section we describe the script `opt_coeffs_v0.2.0.m` that can be found in Appendix.

6.1.1 Functions

For whole optimization a bucket of functions is needed. We define function `lg` performs base-2 logarithm of the argument. It is important element for all information theoretical computations.

```
lg[x_] := Log[2, x]
```

Important part of CF processing is single-tap equalization. Function `\[Alpha]opt` computes optimal inflation coefficient for given SNR (`Pc/Nw`), channel coefficients `h` and chosen HNC map `a`. The function output then minimizes residual lattice misalignment (Mean Square Error (MSE)). Results for different values of SNR are listed in table 6.1.

```
\[Alpha]opt[Pc_, Nw_, a_, h_]:=(Pc Conjugate[h].a )/(Pc (Norm[h])^2 + Nw)
```

SNR[dB]	`h`	`a`	`\[Alpha]opt`
5	[2.9,2.1,0.9]	[1,1,0]	0.359
10	[2.9,2.1,0.9]	[1,1,0]	0.364
20	[2.9,2.1,0.9]	[1,1,0]	0.367

Table 6.1: Optimal inflation coefficients for different values of SNR.

Resulting inflation coefficient is one of the inputs for function `Rif` giving residual interference caused by nonideal single-tap equalization. As we mentioned in theoretical part, contribution such interference is given by MMSE. Function `Norm` in this case performs Euclidean norm of given vector. Again we list obtained result for different levels of noise power in table 6.2.

```
Rif[Pc_, Nw_, a_, h_] := Pc (Norm[\[Alpha]opt[Pc, Nw, a, h] h - a])^2
```

SNR[dB]	`h`	`a`	`Rif`
5	[2.9,2.1,0.9]	[1,1,0]	0.167
10	[2.9,2.1,0.9]	[1,1,0]	0.166
20	[2.9,2.1,0.9]	[1,1,0]	0.166

Table 6.2: Residual interference for different values of SNR.

Now we can use functions `\[Alpha]opt` and `Rif` for evaluating of maximal achievable rate. Residual interference obtained from functions described above contributes to noise level and decreases computation rate which is result of `RcmpC`. Given code is universally applicable for both channels (real and complex). Table 6.3 shows possible results for given SNR.

```
RcmpC[Pc_, Nw_, a_, h_] :=
Max[ lg[Pc/(Rif[Pc, Nw, a, h] + Abs[\[Alpha]opt[Pc, Nw, a, h]]^2  Nw)], 0]
```

SNR[dB]	h	a	Rcmp [bits/dim]
5	[2.9,2.1,0.9]	[1,1,0]	2.27
10	[2.9,2.1,0.9]	[1,1,0]	2.48
20	[2.9,2.1,0.9]	[1,1,0]	2.58

Table 6.3: Residual interference for different values of SNR.

6.2 HNC map optimization

In previous section we only showed functions representing mathematical expressions described in theoretical part of the thesis. Therefore, the whole code seems to be boring and not interesting. However, we now show its importance for the next parts of the script. In section 4.1 we mentioned that in order to obtain good computation rates we must pay attention on HNC map optimization. We also described possible ways to choose HNC map. Determining these maps is performed by code below. Without lost of generality we describe the case of complex channel. At first, we create integer vector raster in close vicinity of channel coefficients and save it to variable `AintsC`. In the next step, the zero vector is removed since its usage has not practical aspects. First way how to determine good map is brute force method. Computation rate is then evaluated for each vector of integer raster individually and results are stored in temporary variable `tempC`. We attempt to find maximal value. Positions of given values represents variable `inds`. If number of positions is only one, optimal map `aoptC` is found. In the case of multiple maps, we choose the one with the smallest distance from channel vector. The second approach is to create map with rationalization. The map `ahQC` is created by remove fractional property, while mutual relation is preserved. The last procedure how to generate good map is channel coefficients rounding. Such map is stored in variable `aoptCR`. Table 6.4 shows generated maps for given channel vector.

```
AintsC = Flatten[Table[{(i-5)+(j-5)I,(k-5) + (l-5)I,(m-5)+(n-5)I},
{i, 10},{j, 10},{k, 10},{l, 10}, {m, 10}, {n, 10}], 5];
AintsC = Delete[AintsC, Flatten[Position[AintsC, {0, 0, 0}]]];
tempC = RcmpC[Pc, Nw, #, h1] & /@ AintsC;
inds = Position[tempC, Max[tempC]];
x = Norm[h1 - Flatten[AintsC[[#]]]] & /@ inds;
Position[x, Min[x]];
aoptC = Flatten[AintsC[[Flatten[inds[[Flatten[Position[x, Min[x]]]]]]]]]
aoptCR = Round[h1]
ahQC = hQ Max[Denominator[hQ]]
```

Channel coefficients	$[3.1 + 0.8j, 1.9 + 1.9j, 1.1 + 2.9]$
Brute force	$[2 - j, 2, 2 + j]$
Rationalization	$[3 + j, 2 + 2j, 1 + 3j]$
Rounding	$[31 + 8j, 19 + 19j, 11 + 29j]$

Table 6.4: HNC maps generated by different methods

Above we showed part of the code generating HNC maps by distinct manner. Results presented in section 4.1 were evaluated by using this.

6.3 Hierarchical Interference Cancellation

Important contribution of the thesis is H-IFC with successive CF. We divide this section into two parts. In the first subsection we describe script `CF-HifcComplex_v1.2.0.m` that was used for H-IFC results evaluation. In the second part of this section we focus on auxiliary map selection - part of `opt_coeffs_v0.2.0.m`.

6.3.1 Functions

Evaluation of results for individual maps was performed by using script described above. Now we demonstrate slight modification of it allowing to generate computation rates after H-IFC procedure. Note that function giving computation rate for given map is still the same. The main difference is in its inputs, resp. in the form of residual interference caused by nonideal lattice inflation. Therefore, we further describe only evaluation of such interference.

The following code represents behavior of one-step recursive form of H-IFC. In order to compute effective channel coefficients `hy` we must determine optimal `\[Beta]opt2` coefficient for appropriate auxiliary map `aifc`. We remind that `\[Beta]opt2` coefficient serves as projector. Then MMSE coefficient is determined for channel `hy` and desired map `a`. Variable `\[Alpha]opt` then influates residual interference `Pifc3[Nw_]`. Obtained results serve as inputs for `RcmpC` shown above.

```
\[Beta]opt2 = Conjugate[aifc].(h - a) /(Norm[aifc])^2
hy = h - \[Beta]opt2 aifc
\[Alpha]opt3[Nw_] := (Pc Conjugate[hy].a )/(Pc (Norm[hy])^2 + Nw)
Pifc3[Nw_] :=Block[{\[Alpha]opt =
       \[Alpha]opt3[Nw]},Pc (Norm[\[Alpha]opt hy - a])^2]
```

In the case of one-step joint H-IFC, we must determine both inflation coefficients `\[Alpha]opt4[\[Gamma]_]`, `\[Beta]opt4[\[Gamma]_]`. We showed that closed form solution can be find by usage of generalized derivative. However it leads to quite difficult system of equations. Thus, closed form solution were determined analytically by using of MATHEMATICA power and obtained solution is then used for `\[Alpha]opt4[\[Gamma]_]`, `\[Beta]opt4[\[Gamma]_]`.

```
A = {{\[Gamma] (Norm[hS])^2+1,-\[Gamma]2Re[Conjugate[hS].aifcS]},
   {\[Gamma] 2 Re[Conjugate[aifcS].hS], -\[Gamma] (Norm[aifcS])^2}};
b = {\[Gamma] 2Re[Conjugate[hS].aS],\[Gamma] 2Re[Conjugate[aifcS].aS]};
Ai = Inverse[A] // Simplify;
Ai.b//Simplify

\[Alpha]opt4[\[Gamma]_] := (\[Gamma] (-Conjugate[aifc].a Conjugate[
        h].aifc + Conjugate[h].a Norm[h]^2))/
(-\[Gamma] Conjugate[aifc].h Conjugate[h].aifc + Norm[h]^2 +
\[Gamma] Norm[h]^4)
\[Beta]opt4[\[Gamma]_] := (-\[Gamma] Conjugate[aifc].h Conjugate[
       h].a + Conjugate[aifc].a (1 + \[Gamma] Norm[h]^2))/
(\[Gamma] Conjugate[aifc].h Conjugate[h].aifc - Norm[h]^2
 (1 + \[Gamma] Norm[h]^2))
```

When we have available scaling coefficients we can compute residual interference caused by nonideal multi-tap equalization. Interference `Pifc4` serves as input for classical computation rate evaluation.

```
Pifc4[Nw_] := Block[{\[Alpha]opt = \[Alpha]opt4[Pc/Nw],
        \[Beta]opt = \[Beta]opt4[Pc/Nw]},
        Pc (Norm[\[Alpha]opt h - \[Beta]opt aifc - a])^2]
```

6.3.2 Auxiliary map selection

In our hierarchical cancellers we use auxiliary HNC maps. The good point is how to choose these maps in order to achieve good performance. In order to obtain results presented in chapter 5 we used such maps that maximize resulting computation rate for desired map. These parts of code are contained in `opt_coeffs_v0.2.0.m`.

In this paragraph we describe map selection for recursive H-IFC. Desired map is represented by variable `aDesC`. For this map computation rate is obtained and stored in variable `RcmpDesC`. Then we select only maps having computation rate higher than `RcmpDesC` and we store them in `sC`. For each map in `sC` effective channel is created by using function `EffChan` and map inducing the best channel with respect to desired map is considered as optimal and is stored in `aoptHifcC`.

```
heC = h1;
ratesC = {};
vectsC = {};
aDesC = {1 + I, 1 + I, 1 + I};
RcmpDesC = RcmpC[Pc, Nw, aDesC, heC]
survC = Select[tempC, # > RcmpDesC &];
sC = {};
sC = FindInds[sC, survC, tempC];
AintsSurvC = AintsC[[#]] & /@ sC;
EffChann[aHat_, aTilde_, h_] := h - \[Beta]opt[aHat, aTilde, h] aHat
temp2C = RcmpC[Pc, Nw, aDesC, EffChann[#, aDesC, heC]] & /@ AintsSurvC;
aoptHifcC = AintsSurvC[[Flatten[Position[temp2C, Max[temp2C]]]]][[1]]
```

In similar manner, we determine optimal auxiliary map for joint H-IFC. Now we attempt to minimize result of function `RifJ` giving residual interference for given channel, desired and auxiliary map. Again, resulting map with the best performance is stored in `aoptHifcC`.

```
RifJ[Pc_, Nw_, aTilde_, a_, h_] :=
  Pc (Norm[\[Alpha]opt4[Pc/Nw, aTilde, a, h] h - \[Beta]opt4[Pc/Nw,
  aTilde, a, h] aTilde - a])^2;

aDesC = {1 + I, 1 + I, 1 + I};
RcmpDesC = RcmpC[Pc, Nw, aDesC, heC]
survC = Select[tempC, # > RcmpDesC &];
sC = {};
sC = FindInds[sC, survC, tempC];
AintsSurvC = AintsC[[#]] & /@ sC;
temp3C = RifJ[Pc, Nw, #, aDesC, h1] & /@ AintsSurvC;
aoptHifcC = AintsSurvC[[Flatten[Position[temp3C, Min[temp3C]]]]][[1]]
```

In table 6.5 there are listed generated HNC maps by code explained above.

Channel coefficients	$[3.1 + 0.8j, 1.9 + 1.9j, 1.1 + 2.9j]$
Desired map	$[1 + j, 1 + j, 1 + j]$
Map for recursive H-IFC	$[-2, -1 - j, -2j]$
Map for joint H-IFC	$[-2 - j, -2 - j, -2 - j]$

Table 6.5: Resulting auxiliary maps for both H-IFC approaches (SNR=5dB).

7 Conclusion

We analyzed CF as possible relaying strategy for WPNC. Analysis involved correct HNC map selection for given channel coefficients. We suggested three possible ways how to select map having good performance. We also demonstrated influence of chosen map to inflation coefficient value. We stated the conclusion that map with perfect behavior at certain level of SNR can cause catastrophic degradation of overall throughput at another level. Therefore, some optimality criteria for this selection must be stated. The form of such criteria is out of scope of the thesis, but it is topic for further research. The most significant contribution of the thesis is H-IFC design with using CF strategy. There was suggested two types of hierarchical interference cancellation - recursive, joint. We performed analysis for real and complex channel and results was compared with standard CF solution. The thesis slightly touches the theoretic aspects of LDLC concept for WPNC. We found that it can make CF realizable. Practical implementation of LDLC is the topic for our future work.

A Compute and Forward

A.1 HNC map selection

Script opt_coeffs_0.2.0.m:

```
(*Init*)
Clear[lg]
lg[x_] := Log[2, x]

SetOptions[Plot, Frame -> True, GridLines -> Automatic, PlotRange -> All,
  PlotStyle -> {Thick}];
SetOptions[ListPlot, Frame -> True, GridLines -> Automatic, PlotRange -> All,
  PlotStyle -> {Thick}];
Needs["PlotLegends`"]

(*Functions*)


(*Computational rates*)

Clear[RcmpC];
Clear[Rcmp];

RcmpC[Pc_, Nw_, a_, h_] :=
  Max[ lg[Pc/(Rif[Pc, Nw, a, h] + Abs[\[Alpha]opt[Pc, Nw, a, h]]^2  Nw)], 0];
Rcmp[Pc_, Nw_, a_, h_] :=
  Max[ lg[Pc/(Rif[Pc, Nw, a, h] + \[Alpha]opt[Pc, Nw, a, h]^2  Nw)], 0];

(*Inflation coefficient*)

Clear[\[Alpha]opt];
\[Alpha]opt[Pc_, Nw_, a_, h_] := (Pc Conjugate[h].a )/(Pc (Norm[h])^2 + Nw
  );

(*Residual interference*)

Clear[Rif];
Rif[Pc_, Nw_, a_, h_] := Pc (Norm[\[Alpha]opt[Pc, Nw, a, h] h - a])^2;

(*Residual interference Joint  H-SD-IFC*)

Clear[RifJ];
RifJ[Pc_, Nw_, aTilde_, a_, h_] :=
Pc(Norm[\[Alpha]optJ[Pc/Nw, aTilde, a, h] h - \[Beta]optJ[Pc/Nw,
 aTilde, a,h] aTilde - a])^2;
```

```
\[Alpha]optJ[\[Gamma]_, aifc_, a_,
  h_] := (\[Gamma] (-Conjugate[aifc].a Conjugate[h].aifc +
      Conjugate[h].a Norm[h]^2))/(-\[Gamma] Conjugate[aifc].h Conjugate[
       h].aifc + Norm[h]^2 + \[Gamma] Norm[h]^4)

\[Beta]optJ[\[Gamma]_, aifc_, a_,
  h_] := (-\[Gamma] Conjugate[aifc].h Conjugate[h].a +
    Conjugate[aifc].a (1 + \[Gamma] Norm[h]^2))/(\[Gamma] Conjugate[
       aifc].h Conjugate[h].aifc - Norm[h]^2 (1 + \[Gamma] Norm[h]^2))

(*Indices finding by value*)

Clear[FindInds];
FindInds[listInd_, List_ /; Length[List] == 0, temp_] := listInd;
FindInds[listInd_, List_, temp_] :=
  FindInds[Flatten[Append[listInd, Flatten[Position[temp, List[[1]]]]]],
   Delete[List, 1], temp];

(*Coefficient beta*)
Clear[\[Beta]opt];
\[Beta]opt[aHat_, aTilde_, h_] :=
  Conjugate[aHat].(h - aTilde) /(Norm[aHat])^2;

(*Parameters*)

(*hR={3.4,2.3,1.6};*)
hR = {2.9, 2.1, 0.9};

(*hC={2.3+0.2I,3.1+1.7I,3.1+1.3I}*)

hC = {3.4 + 0.4 I, 2.3 + 1.7 I, 1.6 + 2.4 I};
hC = {3.1 + 0.8 I, 1.9 + 1.9 I, 1.1 + 2.9 I};
h1 = hR;
hQ = Rationalize[h1];

Pc = 1;
snr = 5;
Nw = Pc/10^(snr/10);
Aints = Flatten[Table[{i - 5, j - 5, k - 5},
 {i, 10}, {j, 10}, {k, 10}], 2];
Aints = Delete[Aints, Flatten[Position[Aints, {0, 0, 0}]]];
AintsC = Flatten[Table[{(i - 5) + (j - 5) I, (k - 5) + (l - 5) I,
 (m - 5) + (n - 5) I}, {i, 10}, {j, 10}, {k, 10}, {l, 10},
 {m, 10}, {n, 10}], 5];
AintsC = Delete[AintsC, Flatten[Position[AintsC, {0, 0, 0}]]];

(*Map optimization*)
tempC = RcmpC[Pc, Nw, #, h1] & /@ AintsC;
inds = Position[tempC, Max[tempC]];
x = Norm[h1 - Flatten[AintsC[[#]]]] & /@ inds;
Position[x, Min[x]];
aoptC = Flatten[
  AintsC[[Flatten[inds[[Flatten[Position[x, Min[x]]]]]]]]]
aoptCR = Round[h1]
ahQC = hQ Max[Denominator[hQ]]
```

```
(*Optimal coefficients for H-SD-IFC - complex valued channel*)

(*Init*)

heC = h1;
ratesC = {};
vectsC = {};

(*Basic iteration*)

aDesC = {1 + I, 1 + I, 1 + I};
RcmpDesC = RcmpC[Pc, Nw, aDesC, heC]
survC = Select[tempC, # > RcmpDesC &];
sC = {};
sC = FindInds[sC, survC, tempC];
AintsSurvC = AintsC[[#]] & /@ sC;

EffChann[aHat_, aTilde_, h_] := h - \[Beta]opt[aHat, aTilde, h] aHat
temp2C = RcmpC[Pc, Nw, aDesC, EffChann[#, aDesC, heC]] & /@ AintsSurvC;
aoptHifcC = AintsSurvC[[Flatten[Position[temp2C, Max[temp2C]]]]][[1]]

(*Recalculation of  the channel*)

ratesC = AppendTo[ratesC, RcmpC[Pc, Nw, aoptHifcC, heC]];
vectsC = AppendTo[vectsC, aoptHifcC]
heC = EffChann[aoptHifcC, aDesC, heC];
tempC = RcmpC[Pc, Nw, #, heC] & /@ AintsC;
inds1C = Position[tempC, Max[tempC]];
x1C = Norm[heC - Flatten[AintsC[[#]]]] & /@ inds1C;
aopt1C = Flatten[
AintsC[[Flatten[inds1C[[Flatten[Position[x1C, Min[x1C]]]]]]]], 2]

(*Optimal coefficients for Joint  H-SD-IFC*)

heC = h1;
ratesC = {};
vectsC = {};

(*Basic iteration)

aDesC = {1 + I, 1 + I, 1 + I};
RcmpDesC = RcmpC[Pc, Nw, aDesC, heC]
survC = Select[tempC, # > RcmpDesC &];
sC = {};
sC = FindInds[sC, survC, tempC];
AintsSurvC = AintsC[[#]] & /@ sC;

temp3C = RifJ[Pc, Nw, #, aDesC, h1] & /@ AintsSurvC;
aoptHifcC = AintsSurvC[[Flatten[Position[temp3C, Min[temp3C]]]]][[1]]
```

A.2 Hierarchical interference cancellation

Script CF-HifcComplex_v1.4.0.m:

```
(*Init*)

lg[x_] := Log[2, x]

SetOptions[Plot, Frame -> True, GridLines -> Automatic, PlotRange -> All,
  PlotStyle -> {Thick}];
SetOptions[ListPlot, Frame -> True, GridLines -> Automatic, PlotRange -> All,
  PlotStyle -> {Thick}];
SetOptions[ListLinePlot, Frame -> True, GridLines -> Automatic,
  PlotRange -> All, PlotStyle -> {Thick}];

Needs["PlotLegends`"]

(*Capacity of interference channel [bit per channel use]*)

(*Gaussian*)

Cpi[p_, n_, pi_] := lg[1 + p/(pi + n)]

(*Lattice code*)

CpiL[p_, n_, pi_] := Max[lg[p/(pi + n)], 0]

(*Parameters*)

marker = "07C";  (* parameter set ID for exporting results *)

(*Channel and codebook*)

Pc = 1; (* codebook power *)
\[Gamma]=Pc/Nw
h0 = {3.4 + 0.4 I, 2.3 + 1.7 I, 1.6 + 2.4 I};  (* large misalignemt *)
h1 = {3.1 + 0.8 I, 1.9 + 1.9 I, 1.1 + 2.9 I};(* small misalignment *)
h = h0;  (* channel coefficients *)

(*Desired coefficients of constellation level HNC map*)
a = {1 + I, 1 + I, 1 + I};
Ma = 3; (* size of the symbol alphabet per dimmension *)

hQ = Rationalize[h]
ahQ = hQ Max[Denominator[hQ]]
ahR = Round[h]
ah = {1 + I, I, I};

(*Aux 2nd HNC map for multi-stage CF*)

ax0 = ah - a (* automatically from lattice matched coeff -
  it guarantees linear dependece for standard multistage CF *)
ax1 = {2 - I, 2, 2 + I}(* manual selection *)
ax = ax0

(*Canceller HNC map*)
```

```
aifc0 = ah - a (* automatically from lattice matched coeff *)
aifc1 = ah (* manual selection *)
aifc = {-2, -1 - I, -I}

\[Alpha]opt0[Nw_] := (Pc Conjugate[h].a )/(Pc (Norm[h])^2 + Nw)

Pifc0[Nw_]:=Block[{\[Alpha]opt=\[Alpha]opt0[Nw]},
Pc (Norm[\[Alpha]opt h - a])^2]

Rcmp0[Nw_] := Block[{\[Alpha]opt = \[Alpha]opt0[Nw],
Pifc = Pifc0[Nw]},CpiL[Pc, Abs[\[Alpha]opt]^2 Nw, Pifc]]

Export[NotebookDirectory[] <> "results_thesis/Rcmp0_" <> marker <>
  ".dat", Table[{SNRdB, Rcmp0[Pc/10^(SNRdB/10)]}, {SNRdB, -10,
     60}], "Table"]
...similarly for other maps.
```

B LDLC design

B.1 Parity check matrix generator

Script parity_check_mat_gen.m:

```
function [ H,H_permuted ] = parity_check_mat_gen( n,d,genSequence)
% function for  LDLC parity check matrix generating
%   n...block length
%   d...degree (assumption of regular LDLC)
%   genSequence...generating sequence

P = zeros(d,n);
for i=1:d
    P(i,:) = randperm(n);
end

c=1;
loopless_columns=0;

while loopless_columns < n
        changed_permutation = 0;

            x_p = find(histc(P(:,c),1:n)>1);

             if isempty(x_p) == 0 % 2-loop test
                     x = find(P(:,c)==x_p(1));
                     changed_permutation=x(1);
             else
                               %4-loop test
                    for j=1:n
                            if c==j
                                continue;
                            end;
                            cmp1 = histc(P(:,c),1:n);
                            cmp2 = histc(P(:,j),1:n);
                            inds = find(cmp1>0 & cmp2>0);

                            if numel(inds)>1
                                inds_x=[];
                                    for k=1:length(inds)
                                inds_x = [ inds_x find(P(:,c)==inds(k))];
                                    end
                                changed_permutation=min(inds_x);
                                break;

                            end
```

```
                    end
             end

        if changed_permutation ~=0
                changed_permutation;
                o=c;
                while c==o
                o = randi(n);
                end
                temp = P(changed_permutation,o);
                P(changed_permutation,o) = P(changed_permutation,c);
                P(changed_permutation,c) = temp;
                loopless_columns=0;
        else
                loopless_columns = loopless_columns +1

        end
        c = c + 1;
        if c > n
            c = 1;
        end


end


H = zeros(n,n);

for i = 1:n
    for j= 1:d
            H(P(j,i),i)=genSequence(j)*(-1)^(randi([0 1]));
    end

end
% permuted parity check matrix for Jacobi method
H_permuted = zeros(n,n);
            for i=1:n
               index = find(abs(H(i,:))==genSequence(1));
               H_permuted(index,:) = H(i,:);
            end
%  H_permuted = H_permuted/det(H_permuted)^(1/n);

end
```

B.2 LDLC encoding

Script `LDLC_decoder.m`:

```
function [ x ] = LDLC_encoder( H_p,b )
% H_p...permuted parity check matrix of given LDLC
% b...data for encoding
% encoding is based on Jacobi method for solving
% H_p*x=b by iterative way
n = size(H_p,1)
D = eye(n).*H_p
L = (-1)*tril(H_p,-1)
```

```
U = (-1)*triu(H_p,1)
D_i = inv(D);

T = D_i*(L+U)
c = D_i*b
eps = 1e-5;
x= zeros(n,1);
i = 0;
err_r = inf;
while err_r>eps
        i=i+1
        x_prev=x;
        x = T*x + c
        err_r = max(abs(x-x_prev))/max(abs(x))


end
end
```

Bibliography

[1] M. Gastpar B. Nazer. Compute-and-forward: Harnessing interference with structured codes. *IEEE Transaction on Information Theory*, 57:6463 – 6486, 2011.

[2] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley & Sons, 2006.

[3] U. Erez and R. Zamir. Achieving 1/2log(1+SNR) on the awgn channel with lattice encoding and decoding. *IEEE Transactions On Information Theory*, 50:2293–2314, 2004.

[4] Ch. Fragouli, J.Y. Le Boudec, and Jörf Widmer. Network coding: An instant primer.

[5] O. Ordentlich, U. Erez, and B. Nazer. The Approximate Sum Capacity of the Symmetric Gaussian K-User Interference Channel. *IEEE Transaction on Information Theory*, 2012. Submitted June 2012.

[6] G. Poltyrev. On coding without restrictions for the AWGN channel. *IEEE Transactions on Information Theory*, 40:409–417, 1994.

[7] N. Sommer, M. Feder, and O. Shalvi. Low density lattice codes. *IEEE Transaction on Information Theory*, 54:1561–1585, 2008.

[8] Jan Sykora and Jan Hejtmanek. Hierarchical Interference Cancellation with Successive Compute & Forward Decoding. 2014. Submitted to PIMRC14 for publication.