AALBORG UNIVERSITY
Department of Mobile Communications

OPTIMIZATION IN DESIGN OF PRECODER AND DECODER FOR COORDINATED ACCESS POINTS SYSTEMS USING MMSE CRITERION IN MEASURED CHANNELS

CARLOS MATALLANA ESPINAR

Supervisors:

Maryam Rahimi

Elisabeth de Carvalho

ABSTRACT

In this project is implemented an iterative algorithm. to determine the linear transmit and receive beamforming filters maximizing the sum-rate of cooperative MIMO system. The channel state information (CSI) of all links is assumed to be known at the transmitters. The algorithm is based on recent papers [5] showing the relation between Weighted Minimum Mean Square Error (WMMSE) and Weighted Sum Rate (WSR) on MIMO Broadcast Channels. This connection allows us to solve the transmit filter with a low complexity algorithm and numerical results shows the algorithm performance is achieved with few iterations. Another contribution of this project is to study the effects when we are working with real measurement data. Real measurement data has been taken in Aalborg (Denmark) and it has tested with the mentioned algorithm. It is shown how real data can be treated with signal processing to test its behavior compared to theoretical channels. Several ways to manipulate real data has been done and the comparison between them for the purpose to realize how it can affect the achievable rate at users. It is concluded and shown that real data cannot achieve the best rate performance due to its imperfections, being the unbalance between branch power ratio the largest detrimental effect.

ACKNOWLEDGEMENT

Firstly, I would like to thank Aalborg University to accept me at university as an Erasmus student and allow me to live this experience, as well as my supervisors, Elisabeth de Carvalho and Maryam Rahimi, for helping and supporting me during the project.

I would also like to thank all the people I have known in Aalborg. These six months of my life would never have been the same without you. I don't want mention names because all of you are special for me and you always will be in my mind.

I want to thank my family, especially my mother and my father, Josefina and Alfonso, for giving me all I have and always believe in me. Also, thanks to my brother Alberto, because he is a good example for me.

Thanks everybody who I have shared my time with. Especially my friends in Mataro: Xavi, Carlos,Ruben, Emilio, Ire, Judit, Nuria, Miona, Dolar and Fede. All the time I have spend with you have been unforgettable, you are part of my life and you always will be. Also thanks to my friends at university, Xavier, Nicolas, Joan and son. I am glad to have shared my university time with you. I want to mention all the time I spent being athletic trainer during my studies. This gave me great times during my studies and thanks to Helena, Cristina, Rosa Maria, Gemma, Gerard, to name a few, for making it possible.

I want to give my best thanks to Judit, my girlfriend. You are the light that illuminates my way; you keep me alive every day; your support and your help has been essential for going ahead; I have no words to describe the feelings I have for you. Without you I am nothing.

Thanks everyone.

TABLE OF CONTENTS

LIST OF ACRONYMS

3G	Third Generation
AWGN	Additive White Gaussian Noise
CDF	Cumulative Distribution Function
IID	Independent and Identically Distributed
MIMO	Multiple Input – Multiple Output
MIMO-BC	Multiple Input – Multiple Output Broadcast Channel
MIMO-IFC	Multiple Input – Multiple Output Interference Channel
MMSE	Minimum Mean Square Error
PDF	Probability Density Function
QoS	Quality of Service
SINR	Signal to Interference plus Noise Ratio
SNR	Signal to Noise Ratio
WGN	White Gaussian Noise
WMMSE	Weighted Minimum Mean Square Error
WSR	Weighted Sum Rate

LIST OF FIGURES

CHAPTER 1

1. INTRODUCTION

1.1. Background

The motivation for transmitting and receiving more data leads research to find efficient solutions to transmit at higher data rate to satisfy the users. In recent years the demand to provide high-speed to support send more data has growth (fast internet access, video streaming, working online, to name just a few) and with that purpose we need more data rate. There are certain restrictions at time to establish a solution for this problem, like the size of the devices or the environment where we are.

We have to take care of the preferences of the user, besides having small terminals; also they desire a total freedom of movements. Thus, wireless technology is proposed. In the last years many systems have appeared to make it possible, Bluetooth, infrared, 3G and so on. The goal of these wireless systems is transmit to the user at the best rate as possible with minimal errors as possible.

New methods are appearing to resolve all these issues, but other challenges rises up such as the available bandwidth, constraint transmit powers or the cost requirements.

One of these proposed solutions is considering more antennas at the transmitters and receivers. This technique is called MIMO system (multiple input – multiple output). In a few time a lot of researchers have studied the advantages of this technique, and the idea expand to the multi-cell and cooperative MIMO which give us more capacity and better service at user, but there are lots of open issues in this field to be studied.

The concept of using multiple antennas at the transmitter and the receiver has lead to higher capacity in wireless systems. In theory the capacity of point-to-point MIMO should increase linearly by increasing the number of using antennas, [1], [2]. However in practice using the idea in the real cellular system does not give us the promised improvement since the system suffers from inter-cell interference and fails in gaining the expected capacity [3].

1.2. Problem definition

Besides the great advantages of the wireless communication, there are still some problems to transmit desired data to the users. One of the big challenges of wireless communication is the interference.

Interference, which is the undesired signal at the receiver, is caused by different reasons such as reusing the frequency in wireless channels. Interference is one of the major problems in wireless communications [4]; it reduces dramatically the achievable rate of the user. Moreover in multi-cell MIMO systems interference comes from the neighboring base station, as it can be seen at figure 1.1. And forbids us to use all the available capacity that channel offers to us.

The mentioned problem leads researcher to use cooperative MIMO to achieve higher capacity in cellular systems by cancelling out the interference from other base stations, in this case several base stations (BSs) in different cells collaborate to transmit signals to users of their coverage. Many different kinds of cooperative MIMO have been studied as an example where base stations are connected to a central service by a high-speed wired [12] or how the receivers can exchange messages between them to help other nodes in decoding [13].
We assume cooperative base stations sharing the knowledge of the channel. The main advantage of cooperative MIMO consists in improving the capacity of the system, coverage and cell edge throughput.

To have better view on this concept we can see in the figure 1.1, each base station cover a geographical area, so to cover an extended area we need more than one base station. The problem of interference comes out when the signal received at one user is not only the signal which has been sent by its base station, the user also receive some interference from the next base stations.

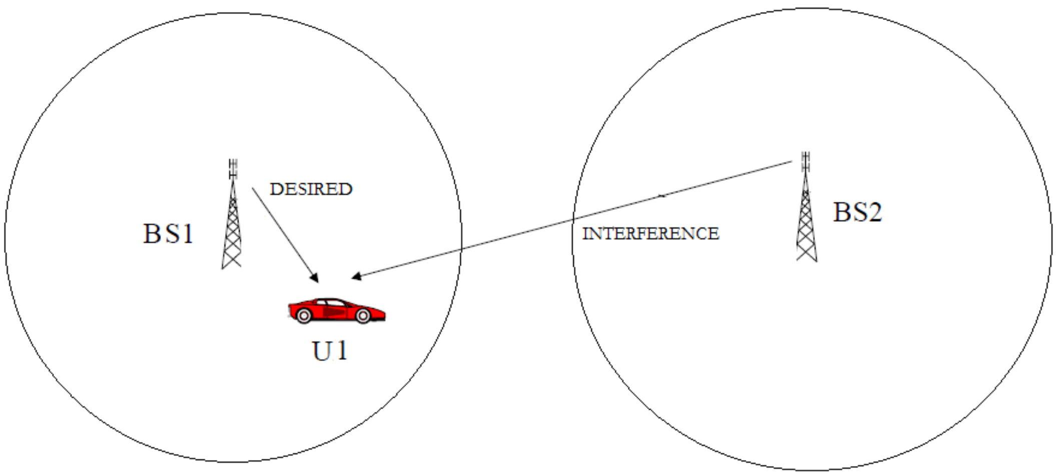


Figure 1.1: Inter-cell interference

1.3. Objectives and motivation

MIMO techniques have been the object of many studies and works, especially in recent years, as it allows for an increased rate for the user. Cooperative MIMO techniques applied to multi-cell interference management have been developed in recent years.

The goal of this work is to study a linear beamforming method allowing multi-cell coordination. Moreover, real measured channels are used in this work to test our technique in real scenarios and combine the theoretical knowledge of cooperative multi-cell MIMO system with the real performance of the system in a realistic propagation environment.

In this work, cooperative MIMO environment is studied, where each user has its own base station. It means that each user receives one desired signal of his base station, also the user receives interference signal from the neighboring base stations. The focus in this work is the application of algorithms to optimize the transmitter and the receiver, and thus get a better rate for the user.

The algorithm which is going to be used will be compared with other well known implementations, such as zero-forcing precoder or matched-filter. Zero-forcing is based on a precoding matrix which is orthogonal to interference channels and leads to cancel the inter-cell interference. On the other hand, matched-filter tries to maximize the gain of the desired signal, but we still have interference at the user.

Concretely the algorithm is based on the relationship between weighted sum-rate and weighted MMSE, which is shown in [5] for a MIMO-BC scenario and treated in MIMO-IFC in [6]. The algorithm gets an optimal precoder and receiver based on MMSE criterion, and try to optimize the relationship among the data input vector and the resulting noise (after receive filter). The algorithm based on weighted MMSE criterion refers that each user has a weight; these weights have to be adjusted to achieve the desired Rate or fit the QoS.

Once the environment has been studied and the algorithms developed, we proceed with the development using real channel measurements which has been collected in the city of Aalborg (Denmark) [7]. Real channel measurements suppose a constraint at time to simulate it and make the comparison among the theoretical results. That is why we focus our simulations in one concrete case, two base stations and two users. Each base station will have two antennas in transmission and the user, one antenna in reception.

1.4. Overview

This work is organized as follows:

Chapter 2

Environment is presented is this chapter. Theoretical background of MIMO channel is shown, concretely a symmetric MIMO channel with K base stations – users.

Chapter 3

In this chapter we described the study about the different algorithms we are going to use. How they apply to the precoder and the receive filter for the environment described in chapter 2.

Chapter 4

Real channel measurements are treated in this chapter. We can see how they have been obtained at Aalborg and the main properties of the measurements will be show.

Chapter 5

Simulations and results about the algorithms shown in chapter 3 are considered in this chapter. Theoretical simulations and real channel simulations are done and also the comparison between them. As we have said in 1.3, we are going to use two antennas at transmitters (base station) and one in reception (users).

Chapter 6

Conclusions are contained in this section.

CHAPTER 2

2. ENVIRONMENT

2.1. System model

In this chapter the environment and scenarios which have been considered in our work are presented. The number of base stations and users are explained. Moreover the scenario for the cooperative base stations and characteristics of the signal are discussed.

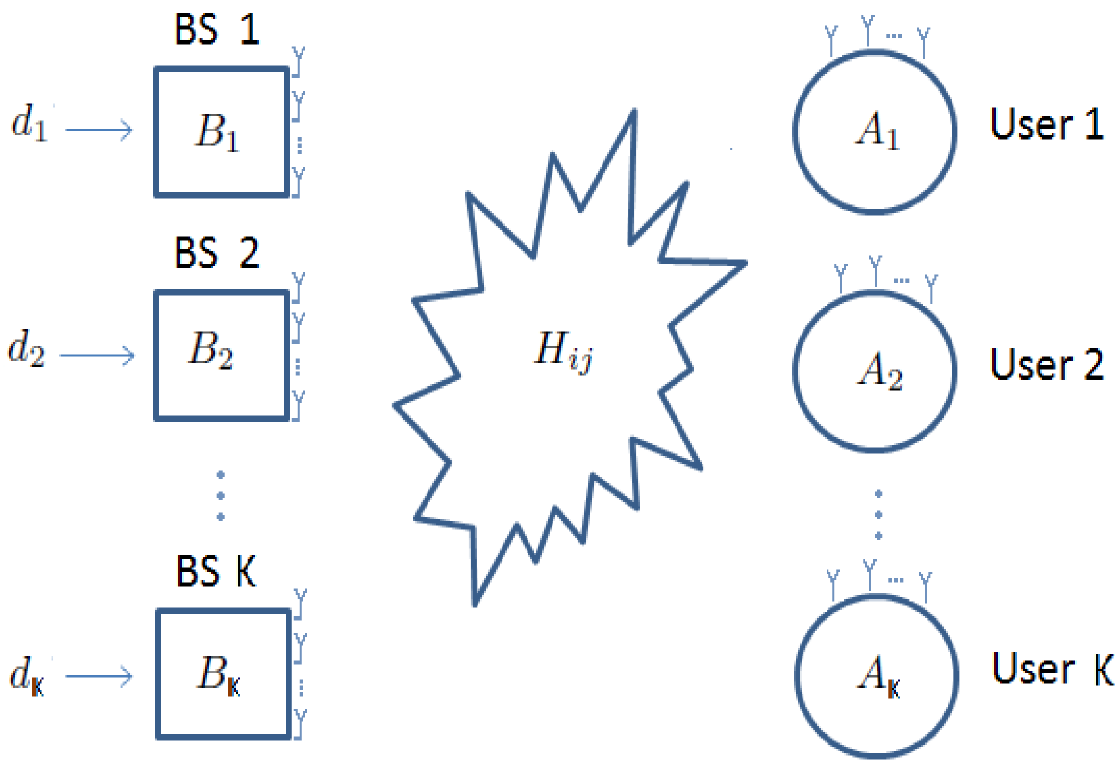


Figure 2.1: System model

As we can see in the figure 2.1, we are treating K base stations and K users. We consider M antennas at each base station and N antennas per user. As we have described before, we have the same number of users and base stations, stations in our assumptions which is called symmetric system. We assume this setup to investigate cooperative base stations in terms of sharing the knowledge of the channel but not in the exchanging the data between base stations. So each base station serves its user. Therefore, each user has its own base station, and the other signals received at the user can be treated as interference.

2.1.1. Transmitted signal

The data stream from base station k has to be sent can be defined as:

$$\boldsymbol{d}_k \in \mathbb{C}^{[N*1]} \tag{2.1}$$

$$\boldsymbol{d}_k = \begin{pmatrix} d_1 \\ \vdots \\ d_N \end{pmatrix} \tag{2.2}$$

$\boldsymbol{d}_k$ is composed by complex values. We consider that our signal follows Gaussian signaling, additionally we assume that we have independent streams; it means that we have the following properties in our signal:

- $E[\boldsymbol{d}_k \boldsymbol{d}_k^H] = \boldsymbol{I}_N$ [1] $\forall\ k$ (2.3)
- $E[\boldsymbol{d}_k \boldsymbol{d}_i^H] = \boldsymbol{0}\ \forall\ k \neq i$ (2.4)

2.1.2. Transmitted filter

Each base station has its own transmit filter $\boldsymbol{B}_k \in \mathbb{C}^{[M*N]}$.

$$\boldsymbol{B}_k = \begin{pmatrix} b_{11} & \cdots & b_{1M} \\ \vdots & \ddots & \vdots \\ b_{1N} & \cdots & b_{NM} \end{pmatrix} \tag{2.5}$$

[1] $\boldsymbol{I}_N$ means the identity matrix with dimension [N*N]

In the matrix we have all the coefficients among all M antennas at the base station and all components of the data stream, N.
These filters are precoding filters and allow us to make a linear filtered version of the transmitted signal. So we obtain the following transmitted vector:

$$\boldsymbol{x}_k = \boldsymbol{B}_k \boldsymbol{d}_k \ , \ \boldsymbol{x}_k \in \mathbb{C}^{[M*1]} \tag{2.6}$$

Each base station transmits the signal to the corresponding user and it is clear that this signal also arrives at the other users as interference signal.

The output of each base station has to satisfy a power constraint. It means that we cannot transmit by more than permitted power. Some papers refer to these power constraints at each transmit antenna, although we can define the power constraint per base station as well.

$$E[||\boldsymbol{x}_k||^2] \leq P_k \tag{2.7}$$

We can develop it using the equation (2.3):

$$E\left[Tr(\boldsymbol{B}_k \boldsymbol{d}_k {\boldsymbol{d}_k}^H {\boldsymbol{B}_k}^H)\right] = Tr(\boldsymbol{B}_k E[\boldsymbol{d}_k {\boldsymbol{d}_k}^H] {\boldsymbol{B}_k}^H) = Tr(\boldsymbol{B}_k {\boldsymbol{B}_k}^H) \leq P_k \tag{2.8}$$

2.1.3. MIMO channels

The channel involved in our system is a MIMO channel between the base station i and the user k.

$$\boldsymbol{H}_{ki} = \begin{pmatrix} h_{11} & \cdots & h_{1N} \\ \vdots & \ddots & \vdots \\ h_{1M} & \cdots & h_{MN} \end{pmatrix} \ , \quad \boldsymbol{H}_{ki} \in \mathbb{C}^{[N*M]} \tag{2.9}$$

The values contained in the matrix shows the gain of the channel between different antenna pairs. Each pair base station – user has a different channel. The channel between the base station k and user k is the channel carrying the desired signal, The channel between base station i and user k where i is different from k carries the inter-cell interference.

$\boldsymbol{H}_{kk}$ → Channel carrying the desired signal

$\boldsymbol{H}_{ki}$ → Channel carrying the interference signal

We assume that the channel follows a Rayleigh distribution.

Equation (2.10) shows the Rayleigh probability density function, and in the figure 2.2 is shown the probability density function for Rayleigh fading samples.

$$f(x|b) = \frac{x}{b^2} e^{\frac{-x^2}{2b^2}} \tag{2.10}$$

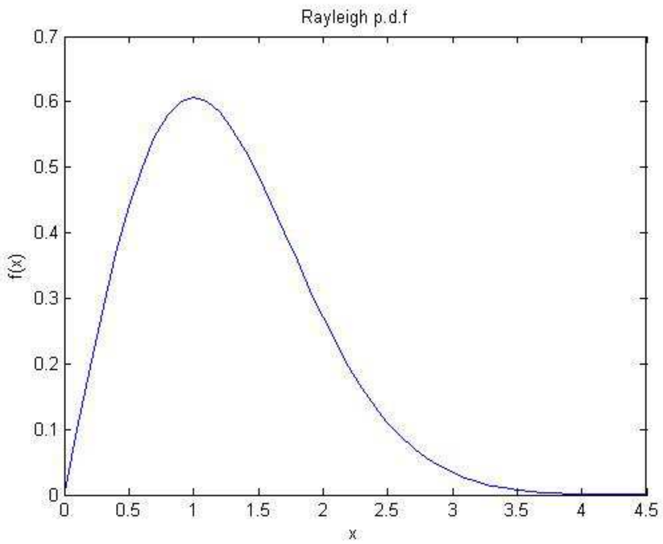


Figure 2.2: Probability density function of Rayleigh fading

To generate Rayleigh distribution channel, we use the following expression in MATLAB:

```
Hki =(1/sqrt(2))*(randn(x,y)+i*randn(x,y))
```
[2]

This expression generates x random values, y times. As it can be seen in figure 2.3 the histogram of the generated values shows us the probability density function of Rayleigh fading distribution which means the channel generated in MATLAB is Rayleigh fading channel.

[2] This expression shows the way to create the MIMO channels coefficients. For the purpose to get the probability density function, we have to take the absolute value of the coefficients. We despise the channel gain.

The samples that we have created have the same pdf as the Rayleigh fading distribution. Concretely we obtain a Rayleigh fading distribution with mean equal to 0 and variance equal to 1, that is because we put a rescale `(1/sqrt(2))` in the expression.

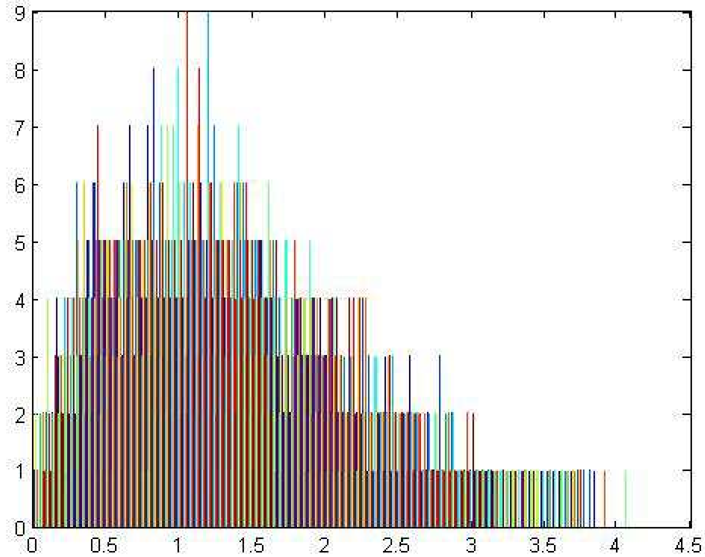


Figure 2.3: Histogram samples created as Rayleigh fading distribution

Theoretical simulations are done with these coefficients, but if we use real measurement to do the simulations we have to check the characteristics of the channel. This will be done in chapter 4 and chapter 5.

2.1.4. Received filter

At the receiver side, each user has a received filter called $\boldsymbol{A}_k \in \mathbb{C}^{[N*N]}$. The received filter allows obtaining a filtered version of the signal from the MIMO channel.

$$\boldsymbol{A}_k = \begin{pmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{1N} & \cdots & a_{NN} \end{pmatrix} \tag{2.11}$$

The objective of this filter is once the signal pass throw the filter, we want to get the data stream which was sent at the beginning.

2.1.5. Noise

Noise is an alteration of the transmitted signal. In our environment the noise is a random signal, which is mixed with our desired signal; $\boldsymbol{n}_k \in \mathbb{C}^{[N*1]}$.

The noise considered in most communication system as White Gaussian Noise (WGN). For our purpose we need to know some properties of the noise. The most important characteristics are the mean, the variance, autocorrelation and the relation with our signal.

Equation (2.12) shows the probability density function of a Gaussian distribution.

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \tag{2.12}$$

The parameter μ is the mean, σ^2 is the variance.

In Figure 2.4 is shown several implementations of Gaussian distribution changing the parameters mean and variance.

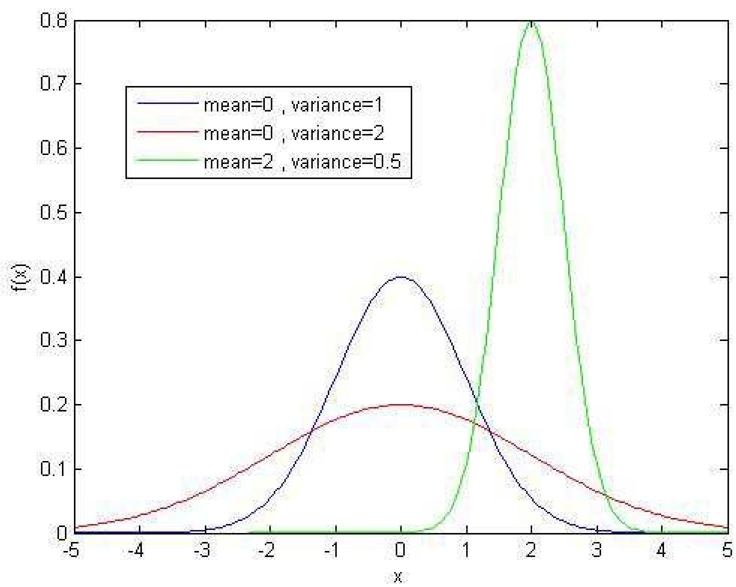


Figure 2.4: Probability density function of Gaussian samples

We take the noise with mean equal to 0 and variance equal to 1; it is going to simplify the mathematical computation in the next chapter.

Noise covariance can be computed as:($\mu = 0, \sigma^2 = 1$):

$$\boldsymbol{R}_{n_k n_k} = E[\boldsymbol{n}_k \boldsymbol{n}_k{}^H] = \boldsymbol{I}_N \quad (2.13)$$

The noise process is independent and identically distributed (iid). At the beginning of this chapter we have considered the transmitted signal as a Gaussian signaling. Furthermore, transmitted signal and noise are independent and de-correlated between them. Thus, they satisfied the following properties:

- $E[\boldsymbol{d}_k \boldsymbol{n}_i^H] = \boldsymbol{0} \ \forall \ k, i$ (2.14)
- $E[\boldsymbol{n}_k \boldsymbol{d}_i^H] = \boldsymbol{0} \ \forall \ k, i$ (2.15)

2.2. Block diagram

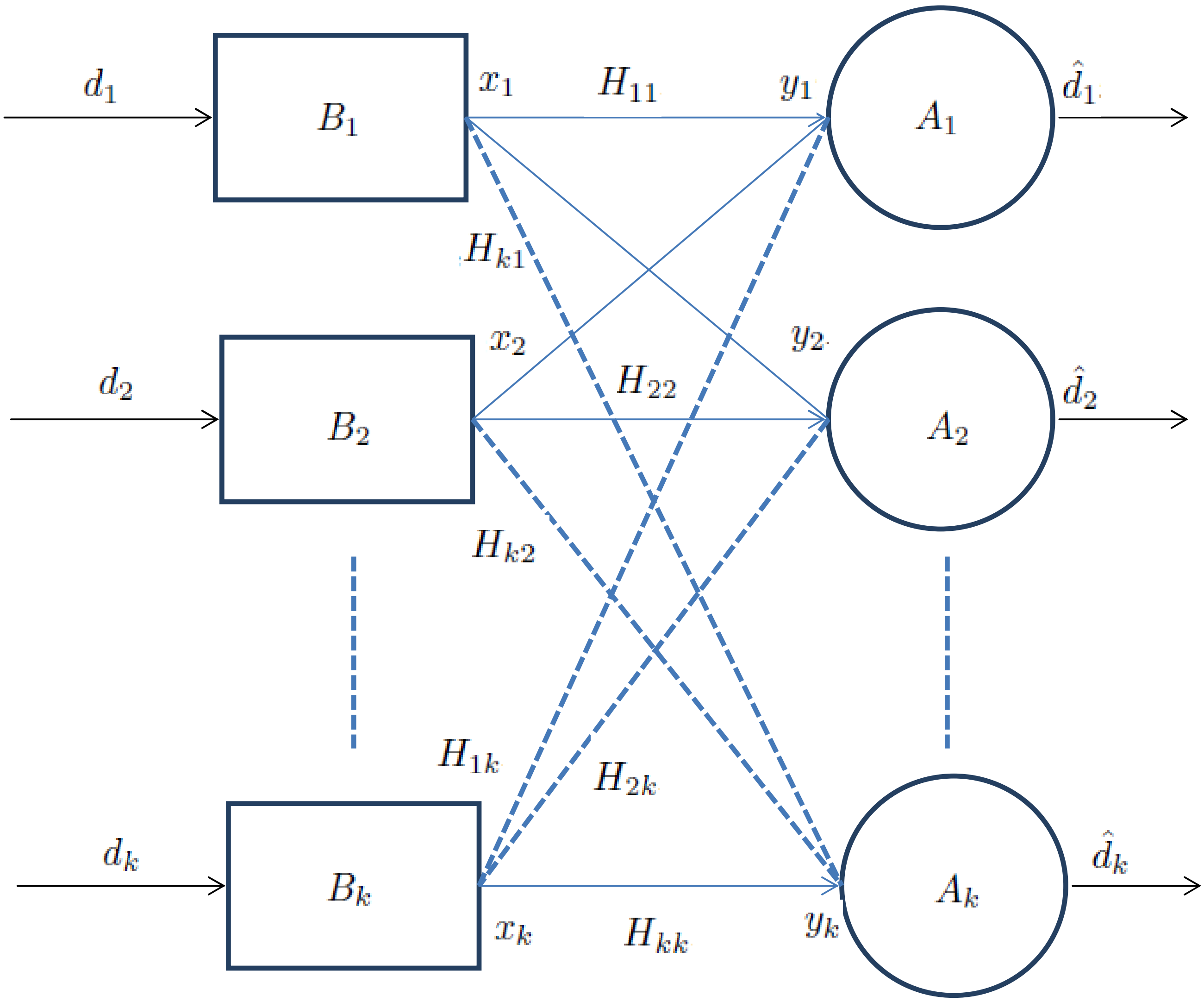


Figure 2.5: Block diagram representing system MIMO KxK

As it can be seen in the figure 2.5, the block diagram shows how the transmitted signal travels through the system.

Firstly, the data stream $\boldsymbol{d}_k$ passes through the transmitted filter $\boldsymbol{B}_k$, so we obtain $\boldsymbol{x}_k$, the linearly filtered version of $\boldsymbol{d}_k$.

$$\boldsymbol{x}_k = \boldsymbol{B}_k \boldsymbol{d}_k \ , \ \boldsymbol{x}_k \in \mathbb{C}^{[M*1]} \tag{2.16}$$

2.2.1. Received signal

Now, $\boldsymbol{x}_k$travels by the channel until arrive at the user. As it can be seen in figure 2.5 user is going to receive one desired signal and $k-1$ interferences. Thus for the user k we have the next receive signal:

$$\boldsymbol{y}_k = \sum_{i=1}^{K} \boldsymbol{H}_{ki}\boldsymbol{x}_i + \boldsymbol{n}_k \quad , \boldsymbol{y}_k \in \mathbb{C}^{[N*1]} \tag{2.17}$$

We can rewrite this equation and separate the desired term and the interference term.

$$\boldsymbol{y}_k = \boldsymbol{H}_{kk}\boldsymbol{x}_k + \sum_{\substack{i=1\\i\neq k}}^{K} \boldsymbol{H}_{ki}\boldsymbol{x}_i + \boldsymbol{n}_k \tag{2.18}$$

If we add equation 2.6, we obtain:

$$\boldsymbol{y}_k = \boldsymbol{H}_{kk}\boldsymbol{B}_k\boldsymbol{d}_k + \sum_{\substack{i=1\\i\neq k}}^{K} \boldsymbol{H}_{ki}\boldsymbol{B}_i\boldsymbol{d}_i + \boldsymbol{n}_k \tag{2.19}$$

Where:

$\boldsymbol{H}_{kk}\boldsymbol{B}_k\boldsymbol{d}_k$ → Desired Signal

$\sum_{\substack{i=1\\i\neq k}}^{K} \boldsymbol{H}_{ki}\boldsymbol{B}_i\boldsymbol{d}_i$ → Interference Signal

$\boldsymbol{n}_k$→ Noise

2.2.2. Rate

Once received signal is known, the achievable rate for each user can be written as:

$$R_k = \log\left(\det\left|\boldsymbol{I}_N + SINR_k\right|\right) \tag{2.20}$$

Where SINR means Signal to Interference plus Noise Ratio at user k. SINR is the value which we can compare the relationship between desired power and interference plus noise power.

$$SINR_k = \frac{P_{desired}}{P_{int}+P_{noise}} = \frac{\boldsymbol{H}_{kk}\boldsymbol{B}_k\boldsymbol{B}_k{}^H\boldsymbol{H}_{kk}{}^H}{\sum_{\substack{i=1\\i\neq k}}^{K} \boldsymbol{H}_{ki}\boldsymbol{B}_i\boldsymbol{B}_i{}^H\boldsymbol{H}_{ki}{}^H + \boldsymbol{I}_N} \tag{2.21}$$

We can call the denominator of the SINR as the effective noise covariance matrix:

$$\boldsymbol{R}_{kk} = \sum_{\substack{i=1\\i\neq k}}^{K} \boldsymbol{H}_{ki}\boldsymbol{B}_i\boldsymbol{B}_i{}^H\boldsymbol{H}_{ki}{}^H + \boldsymbol{I}_N \ , \ \boldsymbol{R}_{kk} \in \mathbb{C}^{[N*N]} \tag{2.22}$$

Finally, the expression that we are going to use in the next chapters for rate at user k is:

$$R_k = \log\left(\det \left| \boldsymbol{I}_N + \frac{\boldsymbol{H}_{kk}\boldsymbol{B}_k{\boldsymbol{B}_k}^H{\boldsymbol{H}_{kk}}^H}{\boldsymbol{R}_{kk}} \right| \right) \tag{2.23}$$

2.2.3. Applying received filters

The last step in block diagram is when received signal at each user goes throw the received filter. If we apply the received filter at the received signal we obtain:

$$\widehat{\boldsymbol{d}}_k = \boldsymbol{A}_k\boldsymbol{y}_k = \boldsymbol{A}_k\boldsymbol{H}_{kk}\boldsymbol{B}_k\boldsymbol{d}_k + \sum_{\substack{i=1 \\ i\neq k}}^{K}\boldsymbol{A}_k\boldsymbol{H}_{ki}\boldsymbol{B}_k\boldsymbol{d}_k + \boldsymbol{A}_k\boldsymbol{n}_k \ , \quad \widehat{\boldsymbol{d}}_k \in \mathbb{C}^{[N*1]} \tag{2.24}$$

2.3. Assumptions

We need to establish some assumptions before start to deploy the algorithms. Below we can see the assumptions that we have done:

- Assuming Gaussian signaling for the transmitted signal
- White Gaussian Noise with mean 0 and variance 1, $\boldsymbol{n}_k \sim \mathcal{CN}(0, \boldsymbol{I}_N)$
- Transmitted signal and noise, are independently and uncorrelated
- Coefficient channels are Rayleigh fading samples with mean 0 and variance equal to 1
- Power constraints as $Tr(\boldsymbol{B}_k \boldsymbol{B}_k{}^H) \leq P_k$, where $P_1 = P_2 = \cdots = P_k = P = 1$

Once we have the coefficients of the channel, we want to see how our algorithm works for different SNR.

$$SNR = \frac{P_d P_H}{P_n} = \frac{P P_H}{\sigma^2} = P_H \qquad (2.25)$$

Where P_H is the power of the channel, $P_H = E[\boldsymbol{h}_{ki} \boldsymbol{h}_{ki}^H]$.

Then, SNR becomes the power of the channel. If we put it in dB,

$$SNR(dB) = 20logSNR(linear) \qquad (2.26)$$

The coefficients which we are going to create and simulate depend directly of the SNR:

```
Hki = 10^(snr(dB)/20)* (1/sqrt(2))*(randn(x,y)+i*randn(x,y))
```

For the simulations with the real measurements we are going to work in a simple environment composed by: two base stations, two users, two antennas at each base station and one antenna at each user (M=2 and N=1).

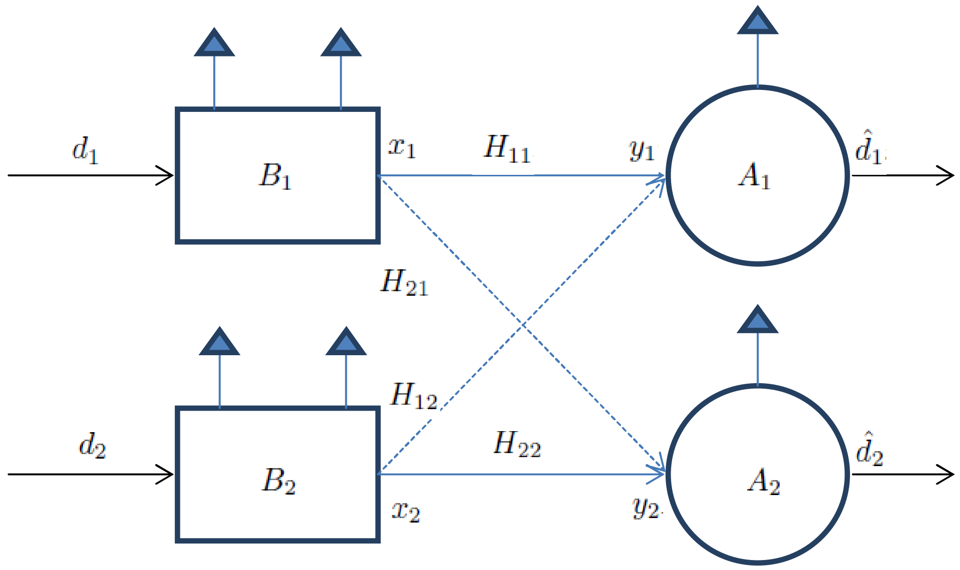


Figure 2.6: 2x2 MIMO system with two antennas at base stations and one antenna at the receiver

CHAPTER 3

3. ALGORITHMS

In this chapter different precoding techniques will be described. The main objective of designing precoder and decoder in this work is developing an algorithm to achieve higher sum rate. As we mentioned in the introduction chapter, ZF pre-coding and matched filter are well known algorithms, because of that we focus more in the Weighted MMSE algorithm in this chapter.

3.1. Zero-forcing pre-coder

Algorithm based on zero-forcing pre-coder gives us the transmit filter allowing to suppress the interference at the received signal. In this algorithm we do not need an interference cancellation procedure at the user. The way to cancel interference is looking for the orthogonal vector to the matrix which is multiplied by interference.

$$\boldsymbol{y}_k = \boldsymbol{H}_{kk}\boldsymbol{B}_k\boldsymbol{d}_k + \sum_{\substack{i=1\\i\neq k}}^{K} \boldsymbol{H}_{ki}\boldsymbol{B}_i\boldsymbol{d}_i + \boldsymbol{n}_k \tag{3.1}$$

We want $\boldsymbol{H}_{ki}\boldsymbol{B}_i = \boldsymbol{0}\ \forall\ i \neq k$ to suppress the interference, which we call $\boldsymbol{B}_k^{\perp}$. Once applied the pre-coding filter we have the following receive signal:

$$\boldsymbol{y}_k = \boldsymbol{H}_{kk}\,\boldsymbol{B}_k^{\perp}\boldsymbol{d}_k + \boldsymbol{n}_k \tag{3.2}$$

Hence the expression for the rate becomes simply:

$$R_k = \log\left(\det\left|\boldsymbol{I}_N + \frac{\boldsymbol{H}_{kk}\,\boldsymbol{B}_k^{\perp}\,{\boldsymbol{B}_k^{\perp}}^{H}{\boldsymbol{H}_{kk}}^{H}}{\boldsymbol{I}_N}\right|\right) \tag{3.3}$$

Note that there are many papers which study this algorithm in a more difficult way trying to join ZF with others strategies for achieving better results. In [8] zero-forcing is used joint by decision feedback at the receivers, [9] is proposed a low complexity algorithm for successive zero-forcing pre-coding, also is shown the combination between successive zero-forcing and dirty paper coding (SZF-DPC).

3.1.1. Zero-forcing with 2 base stations and 2 users M=2, N=1

If we assume the case where we are going to simulate the real measurements, we have two base stations and two users (Figure 2.3.1). Therefore, the received signal at each user is:

$$\boldsymbol{USER\ 1} \rightarrow \boldsymbol{y}_1 = \boldsymbol{H}_{11}\boldsymbol{B}_1\boldsymbol{d}_1 + \boldsymbol{H}_{12}\boldsymbol{B}_2\boldsymbol{d}_2 + \boldsymbol{n}_1 \quad (3.4)$$

$$\boldsymbol{USER\ 2} \rightarrow \boldsymbol{y}_2 = \boldsymbol{H}_{22}\boldsymbol{B}_2\boldsymbol{d}_2 + \boldsymbol{H}_{21}\boldsymbol{B}_1\boldsymbol{d}_1 + \boldsymbol{n}_2 \quad (3.5)$$

The objective is design $\boldsymbol{B}_1^{\perp}$, $\boldsymbol{B}_2^{\perp}$ such that: $\boldsymbol{H}_{12}\,\boldsymbol{B}_2^{\perp} = \boldsymbol{0},\ \boldsymbol{H}_{21}\boldsymbol{B}_1^{\perp} = \boldsymbol{0}$.

For the purpose to get $\boldsymbol{B}_1^{\perp}$, $\boldsymbol{B}_2^{\perp}$, we follow the next steps:

$$\boldsymbol{P}_1^{\perp} = \boldsymbol{I}_M - \frac{\boldsymbol{H}_{21}{}^{H}\boldsymbol{H}_{21}}{\boldsymbol{H}_{21}\boldsymbol{H}_{21}{}^{H}} \quad (3.6)$$

$$\boldsymbol{P}_2^{\perp} = \boldsymbol{I}_M - \frac{\boldsymbol{H}_{12}{}^{H}\boldsymbol{H}_{12}}{\boldsymbol{H}_{12}\boldsymbol{H}_{12}{}^{H}} \quad (3.7)$$

$$\boldsymbol{B}_1^{\perp} = \boldsymbol{P}_1^{\perp}\boldsymbol{H}_{11}{}^{H} \quad (3.8)$$

$$\boldsymbol{B}_2^{\perp} = \boldsymbol{P}_2^{\perp}\boldsymbol{H}_{22}{}^{H} \quad (3.9)$$

Once we have the transmit filters, we have to take care about the power constraint at each transmitter (eq. power constraint), that is why we need to normalize the filters:

$$\overline{\boldsymbol{B}}_1^{\perp} = \frac{\boldsymbol{B}_1^{\perp}}{||\boldsymbol{B}_1^{\perp}||} \quad (3.10)$$

$$\overline{\boldsymbol{B}}_2^{\perp} = \frac{\boldsymbol{B}_2^{\perp}}{||\boldsymbol{B}_2^{\perp}||} \quad (3.11)$$

Note that zero-forcing filter no require an iterative algorithm.

3.2. Matched filter pre-coder

Matched filter tries to maximize the power associated with the desired signal. However, we do not suppress totally the interference at the received signal. Hence, we don't get a good performance with this filter, but it will serve for doing comparisons among the other algorithms.

The way to achieve MF is easy. One more time we have to take care of the power constraint.

For the purpose to maximize the desired signal $\boldsymbol{H}_{kk}\boldsymbol{B}_k\boldsymbol{d}_k$, MF is designed as:

$$\boldsymbol{B}_k^{MF} = \frac{\boldsymbol{H}_{kk}{}^H}{||\boldsymbol{H}_{kk}||} \tag{3.12}$$

Rate has the same expression, but adding the MF:

$$R_k = \log\left(\det\left|\boldsymbol{I}_N + \frac{\boldsymbol{H}_{kk}\boldsymbol{B}_k^{MF}\boldsymbol{B}_k^{MF^H}\boldsymbol{H}_{kk}{}^H}{\sum_{\substack{i=1\\i\neq k}}^{K}\boldsymbol{H}_{ki}\boldsymbol{B}_i^{MF}\boldsymbol{B}_k^{MF^H}\boldsymbol{H}_{ki}{}^H + \boldsymbol{I}_N}\right|\right) \tag{3.13}$$

3.2.1. Matched filter with 2 base station and 2 users M=2, N=1

If we follow the way to achieve transmit filter like above, these are the expressions for the transmit filter for each user:

$$\boldsymbol{B}_1^{MF} = \frac{\boldsymbol{H}_{11}{}^H}{||\boldsymbol{H}_{11}||} \tag{3.14}$$

$$\boldsymbol{B}_2^{MF} = \frac{\boldsymbol{H}_{22}{}^H}{||\boldsymbol{H}_{22}||} \tag{3.15}$$

Like ZF, MF no requires an iterative algorithm.

3.3. Weighted Minimum Mean Square Error (WMMSE)

Firstly MMSE criterion is applied at the receivers. Secondly the relationship between Weighted MMSE and Weighted Sum-Rate (WSR) is explained. At last the algorithm is developed, including all its steps.

3.3.1. MMSE criterion at receive filters

Design of receiver MSE involves the minimization of the difference between the receivers output and the input data vector which we send at the first point. Thus, we have to design the optimal receive filter to achieve the desired signal with the smaller error rate possible. We can write this problem as follows:

$$\boldsymbol{A}_k^{MMSE} = \min_{\boldsymbol{A}_k} E[||\hat{\boldsymbol{d}}_k - \boldsymbol{d}_k||^2] = \min_{\boldsymbol{A}_k} E[||\boldsymbol{A}_k \boldsymbol{y}_k - \boldsymbol{d}_k||^2] \quad (3.16)$$

Using mathematical properties in [10], we can develop the above equation until finding the expression for MMSE receive filters:

$$\boldsymbol{A}_k^{MMSE} = \boldsymbol{B}_{kk}^H \boldsymbol{H}_{kk}^H (\boldsymbol{H}_{kk} \boldsymbol{B}_k {\boldsymbol{B}_k}^H {\boldsymbol{H}_{kk}}^H + \sum_{\substack{i=1 \\ i \neq k}}^{K} \boldsymbol{H}_{ki} \boldsymbol{B}_i {\boldsymbol{B}_i}^H {\boldsymbol{H}_{ki}}^H + \boldsymbol{I}_N)^{-1} \quad (3.17)$$

We can rewrite the expression in a simple way using the effective noise covariance matrix:

$$\boldsymbol{R}_{kk} = \sum_{\substack{i=1 \\ i \neq k}}^{K} \boldsymbol{H}_{ki} \boldsymbol{B}_i {\boldsymbol{B}_i}^H {\boldsymbol{H}_{ki}}^H + \boldsymbol{I}_N \quad (3.18)$$

$$\boldsymbol{A}_k^{MMSE} = \boldsymbol{B}_{kk}^H \boldsymbol{H}_{kk}^H (\boldsymbol{H}_{kk} \boldsymbol{B}_k {\boldsymbol{B}_k}^H {\boldsymbol{H}_{kk}}^H + \boldsymbol{R}_{kk})^{-1} \quad (3.19)$$

Once we have the MMSE receive filters $\boldsymbol{A}_k^{MMSE}$, we can apply it and get the expression for the minimum mean square error for each user:

$$\boldsymbol{E}_k^{MMSE} = E\left[\left|\left|\hat{\boldsymbol{d}}_k - \boldsymbol{d}_k\right|\right|^2\right] = E\left[\left|\left|\boldsymbol{A}_k^{MMSE} \boldsymbol{y}_k - \boldsymbol{d}_k\right|\right|^2\right] = (\boldsymbol{I}_N + \frac{\boldsymbol{H}_{kk} \boldsymbol{B}_k {\boldsymbol{B}_k}^H {\boldsymbol{H}_{kk}}^H}{\boldsymbol{R}_{kk}})^{-1} \quad (3.20)$$

Note that the expression for rate given in (chapter 2) using the equation above, can be written as:

$$R_k = \log(\det |(\boldsymbol{E}_k^{MMSE})^{-1}|) \quad (3.21)$$

3.3.2. WMMSE-WSR

The main objective of WSR in this part of work is achieving better performance for the weighted sum-rate. For that purpose we can write the next minimization problem for finding the transmit filters:

$$\min_{\boldsymbol{B}_1\cdots\boldsymbol{B}_k} \sum_k -u_{Rk}\boldsymbol{R}_k \qquad s.t. \sum_k Tr(\boldsymbol{B}_k{\boldsymbol{B}_k}^H) \leq P_k \tag{3.22}$$

Where u_{Rk} is the weight for user k .

On the other hand, Weighted MMSE wants to minimize the MSE when $\boldsymbol{A}_k^{MMSE}$ is applied. So the optimization problem to find the transmit filters in this case is the next one:

$$\min_{\boldsymbol{B}_1\cdots\boldsymbol{B}_k} \sum_k Tr(\boldsymbol{W}_k\boldsymbol{E}_k) \qquad s.t. \sum_k Tr(\boldsymbol{B}_k{\boldsymbol{B}_k}^H) \leq P_k \tag{3.23}$$

Where $\boldsymbol{W}_k \in \mathbb{C}^{[N*N]}$, is a matrix which is constant and belongs to the user k.

In [5] is shown the simple relation between WMMSE and WSR on MIMO-BC scenario through the comparison of their gradients, this relationship is also demonstrated in MIMO-IFC scenario in [6].

$$\boldsymbol{W}_k = u_{Rk}(\boldsymbol{E}_k^{MMSE})^{-1} \tag{3.24}$$

We use this relation to find the optimal transmit filters which minimize the WMMSE.

The corresponding Lagrangian associated with the WMMSE optimization problem can be written as:

$$F(\boldsymbol{B}_k, \lambda_k) = \sum_{k=1}^{K} Tr(\boldsymbol{W}_k\boldsymbol{E}_k) + \sum_{k=1}^{K} \lambda_k[Tr(\boldsymbol{B}_k{\boldsymbol{B}_k}^H) - P_k] \tag{3.25}$$

Considering the Karush–Kuhn–Tucker, the local optimum of the transmit filters must satisfy the following properties:

- $\frac{\partial F}{\partial B_k^*} = 0, \quad k \in \{1, \dots, K\}$
- $\lambda_k[Tr(\boldsymbol{B}_k{\boldsymbol{B}_k}^H) - P_k] = 0, \quad k \in \{1, \dots, K\}$

- $Tr\left(\boldsymbol{B}_k {\boldsymbol{B}_k}^H\right) - P_k \leq 0, \quad k \in \{1, \dots, K\}$
- $\lambda_k \geq 0, \quad k \in \{1, \dots, K\}$

If we do the partial derivate of F with respect to $\boldsymbol{B}_k^*$, we obtain the expression for calculating the optimal transmit filters:

$$\boldsymbol{B}_k = \left(\sum_{i=1}^{K} {\boldsymbol{H}_{ki}}^H {\boldsymbol{A}_i}^H \boldsymbol{W}_i \boldsymbol{A}_i \boldsymbol{H}_{ki} + \lambda_k \boldsymbol{I}_M\right)^{-1} {\boldsymbol{H}_{kk}}^H {\boldsymbol{A}_k}^H \boldsymbol{W}_k \quad (3.26)$$

3.3.2.1. Lagrange Multipliers

For the purpose to get the $\boldsymbol{\lambda}_k$ values, we are going to follow the method presented in [11]. $\boldsymbol{\lambda}_k$ has to satisfy KKT conditions and they will be updated in each iteration of the algorithm.

If we take the next definition:

$$\boldsymbol{T}_k(n+1) = \sum_{i=1}^{K} {\boldsymbol{H}_{ki}}^H {\boldsymbol{A}_i}^H (n+1) \boldsymbol{W}_i(n+1) \boldsymbol{A}_i(n+1) \boldsymbol{H}_{ki} \ , \boldsymbol{T}_k \in \mathbb{C}^{[M*M]} \quad (3.27)$$

Now, the expression for the optimal transmit filter can be written as:

$$\boldsymbol{B}_k = (\boldsymbol{T}_k(n+1) + \lambda_k \boldsymbol{I}_M)^{-1} {\boldsymbol{H}_{kk}}^H {\boldsymbol{A}_k}^H (n+1) \boldsymbol{W}_k(n+1) \quad (3.28)$$

To satisfy the power constraint λ_k can be searched as:

$$Tr\left(\boldsymbol{B}_k {\boldsymbol{B}_k}^H\right) = \boldsymbol{P}_k \quad (3.29)$$

To achieve an expression to calculate the $\boldsymbol{\lambda}_k$, we are going to develop the equation above. Firstly, we will do the Singular Value Decomposition (SVD) of $\boldsymbol{T}_k(n+1)$.

$$\boldsymbol{T}_k(n+1) = \boldsymbol{U}_k(n+1) \boldsymbol{\Lambda}_k(n+1) \boldsymbol{U}_k^H(n+1) \quad (3.30)$$

Where $\boldsymbol{U}_k(n+1) \in \mathbb{C}^{[M*M]}$ is an orthogonal[3] matrix which contains the eigenvectors of $\boldsymbol{T}_k(n+1)$, and $\boldsymbol{\Lambda}_k(n+1) \in \mathbb{C}^{[M*M]}$ is a diagonal matrix with the eigenvalues of $\boldsymbol{T}_k(n+1)$.

$$Tr\left(\boldsymbol{B}_k\boldsymbol{B}_k{}^H\right) = Tr(\boldsymbol{W}_k^H(n+1)\boldsymbol{A}_k(n+1)\boldsymbol{H}_{kk}\boldsymbol{U}_k(n+1)(\boldsymbol{\Lambda}_k(n+1)+\lambda_k\boldsymbol{I}_M)^{-1}\boldsymbol{U}_k^H(n+1)\boldsymbol{U}_k(n+1)$$
$$(\boldsymbol{\Lambda}_k(n+1)+\lambda_k\boldsymbol{I}_M)^{-1}\boldsymbol{U}_k^H(n+1)\boldsymbol{H}_{kk}{}^H\boldsymbol{A}_k{}^H(n+1)\boldsymbol{W}_k(n+1)) \tag{3.31}$$

$$= Tr((\boldsymbol{\Lambda}_k(n+1)+\lambda_k\boldsymbol{I}_M)^{-2}\boldsymbol{U}_k^H(n+1)\boldsymbol{H}_{kk}{}^H\boldsymbol{A}_k{}^H(n+1)\boldsymbol{W}_k(n+1)\boldsymbol{W}_k^H(n+1)\boldsymbol{A}_k(n+1)\boldsymbol{H}_{kk}\boldsymbol{U}_k(n+1)) \tag{3.32}$$

Taking $\boldsymbol{G}_k(n+1) = \boldsymbol{U}_k^H(n+1)\boldsymbol{H}_{kk}{}^H\boldsymbol{A}_k{}^H(n+1)\boldsymbol{W}_k(n+1)\boldsymbol{W}_k^H(n+1)\boldsymbol{A}_k(n+1)$
$\boldsymbol{H}_{kk}\boldsymbol{U}_k(n+1)$, $\boldsymbol{G}_k \in \mathbb{C}^{[M*M]}$ (3.33)

Finally the expression to get Lagrange multipliers can be written as:

$$Tr\left(\boldsymbol{B}_k\boldsymbol{B}_k{}^H\right) = Tr((\boldsymbol{\Lambda}_k(n+1)+\lambda_k\boldsymbol{I}_M)^{-2}\,\boldsymbol{G}_k(n+1)) =$$
$$= \sum_{i=1}^{M}\frac{G_k^{[ii]}(n+1)}{(\lambda_k+\Lambda_k^{[ii]}(n+1))^2} = \boldsymbol{P}_k \tag{3.34}$$

Note that Lagrange multipliers can be searched solving the above equation. If the solution does not exist, Lagrange multipliers will be 0. If that fact happens, we have to rescale $\boldsymbol{B}_k$ in order to preserve the power constraint.

[3] Orthogonal matrices satisfy the next property: $\boldsymbol{U_k}\boldsymbol{U}_k^H = \boldsymbol{I}_M$.

3.3.3. Algorithm WMMSE

The steps of the algorithm are described below:

1. Initialize B_k^{init}[4]
2. Set n = 0
3. Update $\boldsymbol{A}_k^n$, given $\boldsymbol{B}_k^{n-1}$ using (equation for $\boldsymbol{A}_k^{MMSE}$)
4. Update $\boldsymbol{W}_k^n$, given $\boldsymbol{B}_k^{n-1}$using (equation for $\boldsymbol{W}_k = u_{Rk}(\boldsymbol{E}_k^{MMSE})^{-1}$ and $\boldsymbol{E}_k^{MMSE}$)
5. Update $\boldsymbol{\lambda}_k^n$, given $\boldsymbol{A}_k^n$, $\boldsymbol{W}_k^n$ using (equation for $\boldsymbol{\lambda_k}$)
6. Update $\boldsymbol{B}_k^n$, given $\boldsymbol{\lambda}_k^n, \boldsymbol{A}_k^n, \boldsymbol{W}_k^n$, using (equation for B_k)
7. Repeat steps 3-6 until convergence (or a predefined number of iterations)

3.3.4. WMMSE with 2 base stations and 2 users M=2, N=1

As zero-forcing and matched filter, WMMSE has been developed to work in our specific environment as it can be seen in figure 2.6.

Some assumptions have been taken to implement WMMSE algorithm:

- u_{Rk} is equal to 1 for each user
- Convergence of the algorithm have been tested [see Chapter 5] and for simulating real channel, 20 iterations have been taken

[4] Depends of the initialization of $\boldsymbol{B}_k$ we get different minimum local optimization as is shown in [5]. Therefore, the algorithm will be shown for different initialization of $\boldsymbol{B}_k$ in chapter 5.

CHAPTER 4

4. REAL MEASUREMENT DATA

In this chapter real measurement data is explained, studied and analyzed. Measurement setup is explained in depth in [7], [14] and [15].

4.1. Configuration of the measurements

The measurements have been done in Aalborg (Denmark). The setup has consisted in installing two base stations in different places of Aalborg city center (figure 4.1 and 4.2).

In the next table (4.1) it can be seen the main properties of the base station locations, as well as the height, distance to user and number of antennas available per each.

Figure 4.2: Location base station 1

Figure 4.1: Location base station 2

	Height [m]	Distance to user [m]	Nº of transmit antennas at low band [776MHz]	Nº of transmit antennas at high band [2.3 GHz]
BS 1	13	150	2	2
BS 2	~ 60	500	2	2

Table 4.1: Base station overview

As it can be seen in the table, both base stations operate in low and high band frequency. The measurements have been done simultaneously in both frequency bands.

In reference to the users, four handsets are testing at the same time which each one has two receive antennas. The measurements have a rate of 60Hz (17 ms), and the frequency which receive signal has been sampled at 400MHz.

For the purpose to do the measurements, each handset have optical fiber for connecting to the sounder, it makes the measurements more accurate. The optical unit to measure the signal receive is explained in [14]. Also the authors of [14] implement a cover for each handset with intend to get closer to real handsets which are been used nowadays. This recover avoids getting more interference caused by the user as well. Following with the user interference, handsets have been measured with different kind of grips to get closer to real scenarios (figure 4.3).

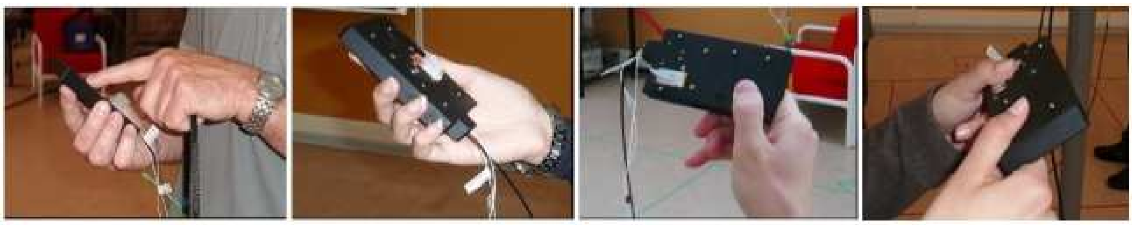


Figure 4.3: Types of grids

Another parameter has been considered in this measurement campaign is the movement of the users. Measurements have been done following two different movements pattern: local average and straight line in a square, it can be seen in figure 4.4. Measurements have been done 1200 times during the movements. Also the measurements have been done in two different cases: free space (where around the handset there are nothing for causing loss at receive signal) and with different users holding the handsets.

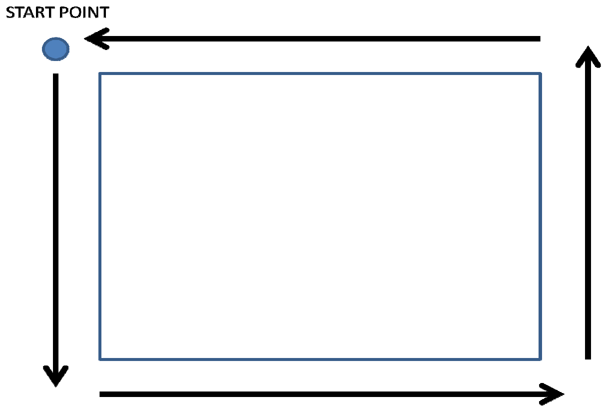


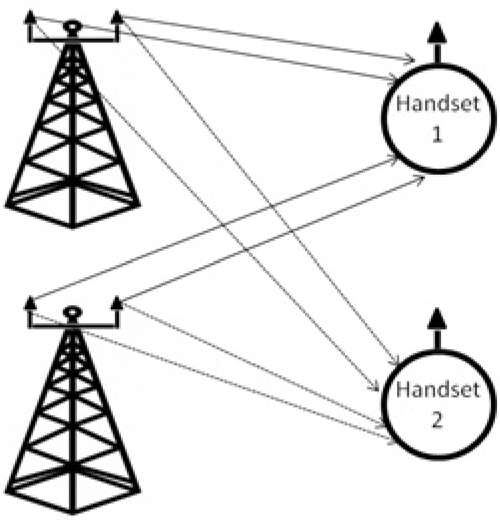
Figure 4.4: Handset movement

The objective of these different situations is study the influence of the Body Loss at the receive signal and the effects of movements. Results are shown in [15].

4.1.1. Scenario considered

The scenario which we are going to investigate in this work is composed by two base stations serve two handsets. Each base station has two antennas and one antenna per each handset. The scenario is shown in figure 4.5. We take the antennas working in high band (2.3 GHz) and in free-space mode, with straight line movement in a square, see figure 4.4.

The objective of this work is comparison the performance of the algorithms describes in chapter 3, in one hand with theoretical channels and on the other hand using real measurement data. The comparison will be done on terms of weighted sum rate.



igure 4.5: Real measurement data scenario

Hence, the matrix $\boldsymbol{H}_{ki}$ between the user k and the base station i has the next shape:

$$\boldsymbol{H}_{ki} = (h_{11} \;\; h_{12}) \tag{4.1}$$

Where h_{11}means the coefficient between antenna 1 at base station i and the antenna at user k. Where h_{12}means the coefficient between antenna 2 at base station i and the antenna at user k.

In total we have four different links between base station and users and the matrices are as:

$$\boldsymbol{H}_{11} = (h_{11} \;\; h_{12})$$

$$\boldsymbol{H}_{12} = (h_{11} \;\; h_{12})$$

$$\boldsymbol{H}_{21} = (h_{11} \;\; h_{12})$$

$$\boldsymbol{H}_{21} = (h_{11} \;\; h_{12})$$

4.2. Measurements analyzing

In this section, the data about the real measurements channels will be analyze. The way to know if the data has good properties will be through a comparison with the properties of a theoretical channel.

4.2.1. Channel normalization

We want to see the performance of ours algorithms in function of SNR. Therefore, channel normalization has been done to may choose an arbitrary SNR. However, there are several ways to do the normalization, each one with different features. Depends on the features which had been kept for the channel, the performance will change; it means that all properties in channel have effects to compute the rate.

4.2.1.1. Normalization to a reference handset

This kind of normalization allows keeping the parameters which have effect on capacity performance, such as different levels of power from different base stations, effects of fading, correlation, branch power and the differences in efficiency of each terminal. This normalization has been chosen to preserve all possible characteristics of the channel, because each one can influence in the calculation of rate.

One handset is taken and the mean power from all its transmitted and received antennas from one base station is computed. Then all the coefficients of all handsets are normalized by the square root of this mean power.

4.2.1.2. Normalization to each different link

In this case, each link is normalized by its own transmit and received antennas. It means that branch power ratio is lost. However, the effect of correlation is preserved.

The mean power of each link is computed and each link is normalized by the square of its mean power, respectively.

$$h_{i,j}^{norm}(p) = \frac{h_{i,j}(p)}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}|h_{i,j}(p)|^2}} \tag{4.2}$$

p means the different 1200 positions where the data have been taken.

4.2.1.3. Normalization removing cross-correlation effects

This normalization is the same like the normalization described in 4.2.1.1, but now the effect of the correlation is removed. The value of the correlation we care is in 0, it shows us how the signals are equal to each other. For the purpose to remove this effect, we move in time the coefficients of links to get more randomly values and then get almost 0 in the correlation between links. It will be shown which links causes the correlations and we have to move for removing it.

$$h_{i,j}^{norm}(p) = h_{i,j}^{norm}(p - lag) \tag{4.3}$$

lag is the delay we put in the links.

Channels normalizations have been done for all the data that it is going to be used.

4.2.2. PDF and Cumulative Distribution Function (CDF)

As it was mentioned in chapter 2, the channel with Rayleigh fading distribution is assumed.

The main properties to know if measured channel has Rayleigh fading distribution is checking the probability density function (PDF) and Cumulative distribution function (CDF). To do that investigation we should know the pdf and cdf of Rayleigh and Rician which can be seen in figure 4.6 and 4.7.

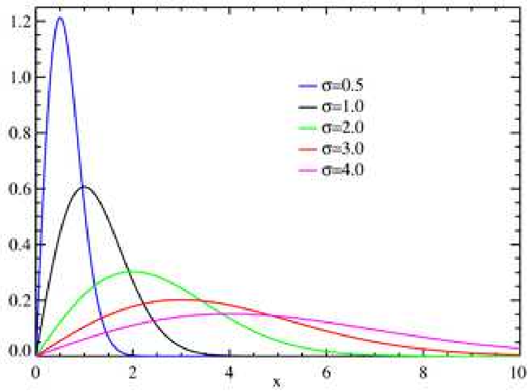


Figure 4.6 : PDF Rayleigh distribution

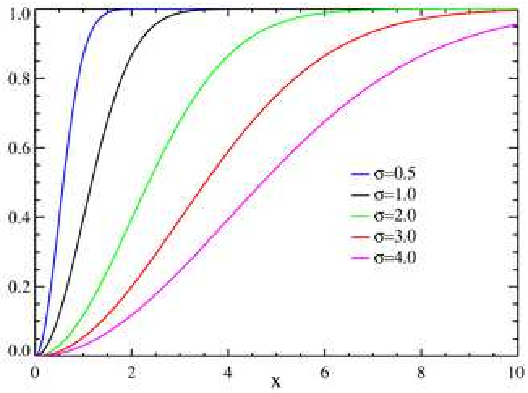


Figure 4.7: CDF Rayleigh distribution

In communications if link with line of sight (LOS) is treated, instead of Rayleigh fading distribution, Rician fading distribution is used. The pdf and cdf of Rician fading samples are shown in figures 4.8 and 4.9.

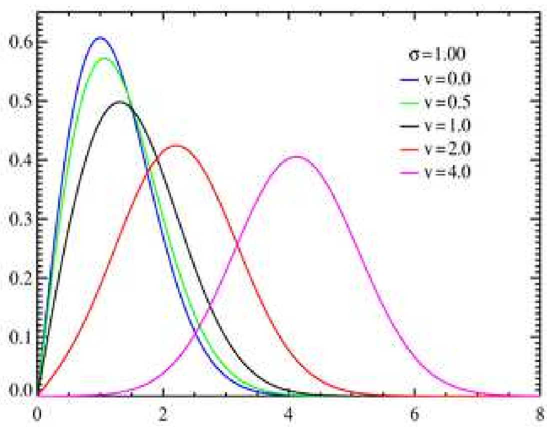


Figure 4.8: PDF Rician distribution

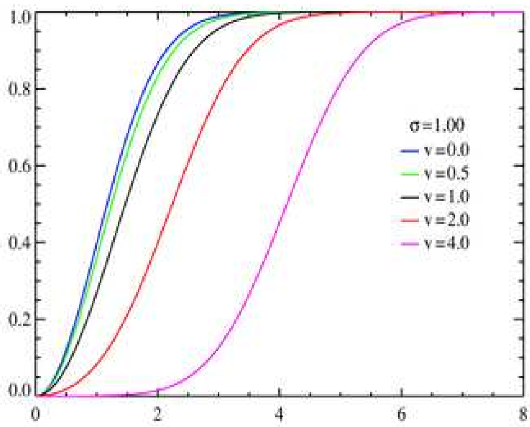


Figure 4.9: CDF Rician distribution

4.2.3. Auto-correlation and cross-correlation

Correlation determines the degree of similarity between two signals. In a theoretical background, if one sequence is taken with numbers totally randomized and independently, auto-correlation has to be 1 in position 0 (no displacement) and 0 in the others positions. If another peak is found in the correlation, the distance between peaks is taken to be the fundamental period of the signal.

For example, if one sequence of numbers which following a Gaussian distribution is taken, the auto-correlation can be seen in the figure 4.10.

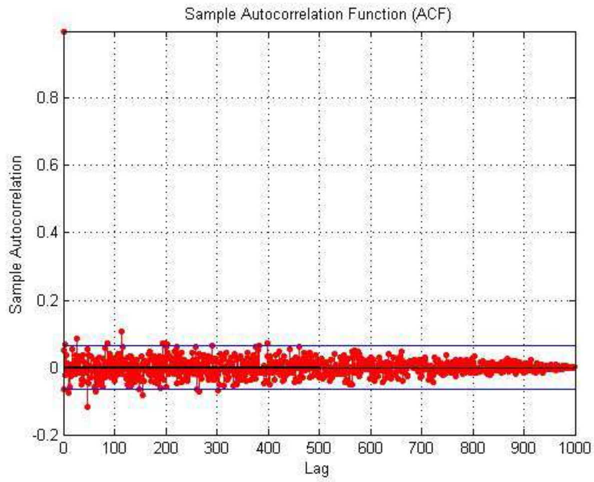


Figure 4.10: Auto correlation examples between two random Gaussian samples

As it can be seen in figure 4.10 there is a peak at position 0, it means that the samples are the same when the samples does not have any delay between them. As it is shown in the figure above, lag can be identified as the delay between samples. Hence, if one sample is moved respect itself (increase lag) 0 is the expected value.

Definition of cross correlation which is shown in the figures is:

$$R_{xy}(lag) = \begin{cases} \sum_{n=0}^{N-|lag|-1} \boldsymbol{x}_{n+m}\boldsymbol{y}_n^* & m \geq 0 \\ R^*{}_{yx}(-lag) & m < 0 \end{cases} \tag{4.4}$$

And the function using in Matlab also normalizes the sequence, so the highest value in autocorrelations can be 1. Auto correlation has the same expression than cross correlation but with the same signal $R_{xx}(lag)$, it is shown for $m \geq 0$.

Another example can be done taking two different sequences. Where two independent signals have been compared. The sequences have been created as Gaussian probability and they are totally independent between them. The correlation between them is close to 0, as it can see in figure 4.11:

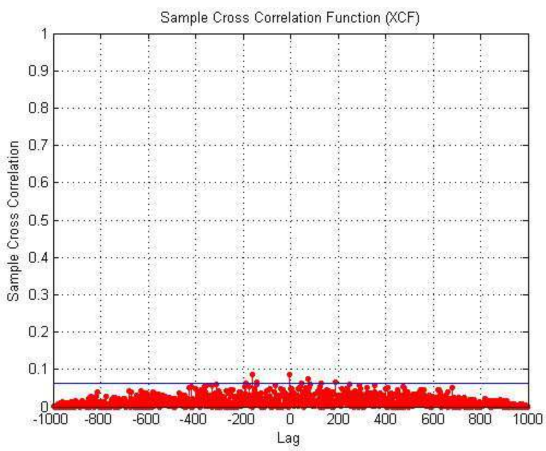


Figure 4.11: Cross correlation between two random Gaussian samples

This is the expected result for the correlation between two random sequences. There is no period, cycle or similitude between the sequences.

The value at $lag = 0$ is the most relevant value in cross-correlation because it determines the similitude between signals at the same time.

4.2.4. Power Delay Profile (PDP)

PDP gives us the intensity of the signal received at the handset in function of the delay time. Delay time is the difference among receive signals. Signal sent can be submitted by some effects like reflection, diffraction and so on. Hence the transmit power arrives at the user at different times. A scheme of PDP can be seen in the next figure:

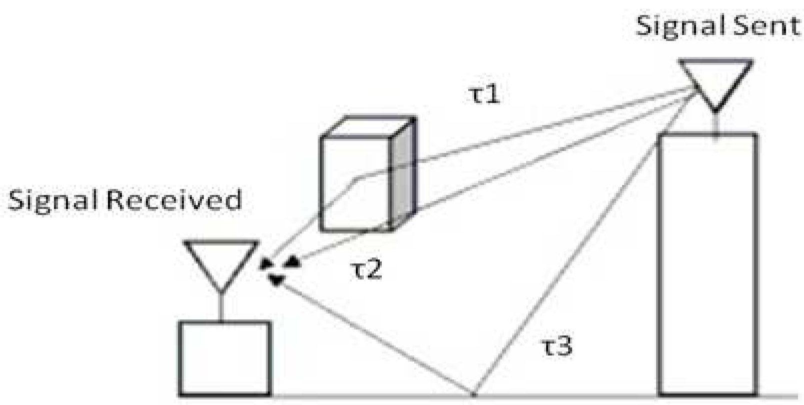


Figure 4.12: Power delay profile scheme

The shape expected of PDP is shown in the figure below:

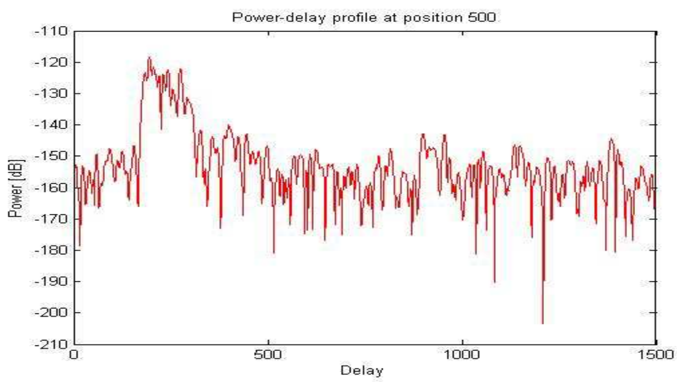


Figure 4.13: Example of PDP at position 500

4.2.5. Gain

The gain informs how good the channel is. If the base station is near to the user, more gain will be available at user.

Next figure shows a typical gain for one user with one antenna and two base stations with one antenna each. One base station is near to the user and the other one is far to the user.

As it can be seen in figure 4.14, it is clear that h1 belongs to the nearest base station and h2 to the far one:

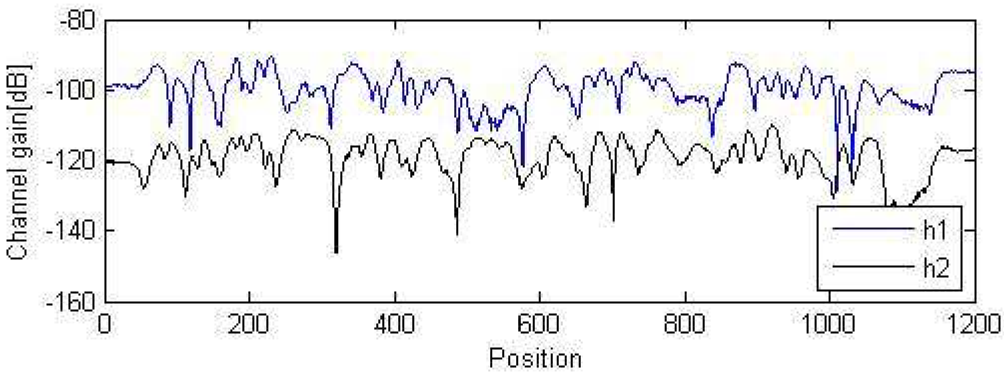


Figure 4.14: Example of channel coefficients gain

4.2.6. Probplot Rayleigh

This function in Matlab (probplot) serves to determine if samples are distributed as a Rayleigh distribution. The samples which have been tested are the blue marks, and if they follow Rayleigh distribution will coincide with the black line (theoretical Rayleigh distribution).

As an example is tested a Rayleigh distribution like the described in 2.1.3:

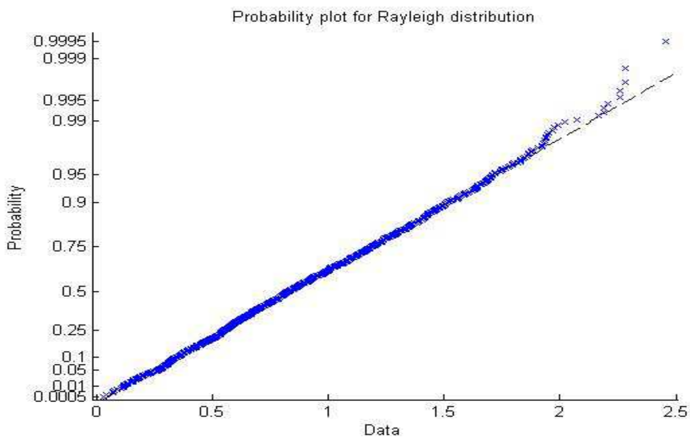


Figure 4.15: Rayleigh distribution test

CHAPTER 5

5. SIMULATIONS

In this chapter simulation results from two different stages will be shown. First the results from WMMSE algorithm is shown while the channels generated as i.i.d Rayleigh. From now on this assumption will be called theoretical simulation and the results achieved by this assumption are theoretical results. The second stage is implementing real measured channels in the WMMSE algorithm to compare and analyze its behavior of the cooperation in BSs in real propagation environment.

It should be noted in all simulation done by real channels while Power Delay Profile is shown the first antenna at base station one and the antenna at user one have been assumed.

5.1. Theoretical results

This section evaluates the performance of the theoretical algorithms for a MIMO scenario 2x2, with two antennas at base stations and one antenna at the users.

5.1.1. Convergence

As explained before the algorithm needs a starting point to run. So an initialization for the algorithm has to be chosen; in this simulation the initial transmitted filter will be the zero-forcing filter $B_k^{init} = B_k^{ZF}$. The convergence has been tested with different SNR values and as it can be seen in figure 5.1, 5.2 and 5.3, convergence is achieved in a few iterations. Due to a fast convergence which has the algorithm, 20 iterations have been considered for the rest of the work.

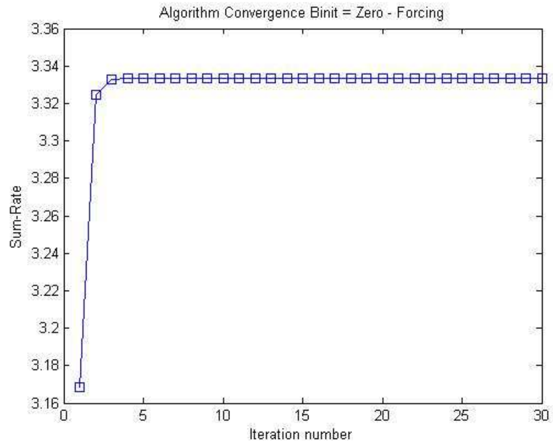


Figure 5.1: Converges properties for randomly cannel realization with SNR = 0dB

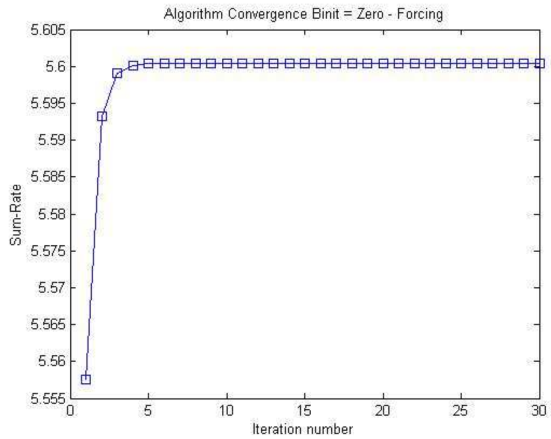


Figure 5.2: Converges properties for randomly cannel realization with SNR = 10dB

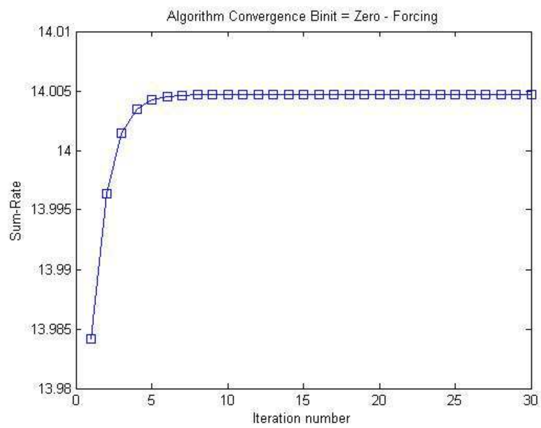


Figure 5.3: Converges properties for randomly cannel realization with SNR = 20dB

5.1.2. Performance

As we mention on chapter 3, WMMSE algorithm is non-convex and depends which initialization we choose, we get different local minimums for the algorithm. Therefore, the algorithm has been simulating with different transmit filters initialization:

- $\boldsymbol{B}_k^{init} = \boldsymbol{I}_{N*M}$
- $\boldsymbol{B}_k^{init} = \boldsymbol{B}_k^{MF}$
- $\boldsymbol{B}_k^{init} = \boldsymbol{B}_k^{ZF}$
- $\boldsymbol{B}_k^{init}$ = Using 10 random initialization for $\boldsymbol{B}_k^{init}$ and choose the best one which gives us the best performance

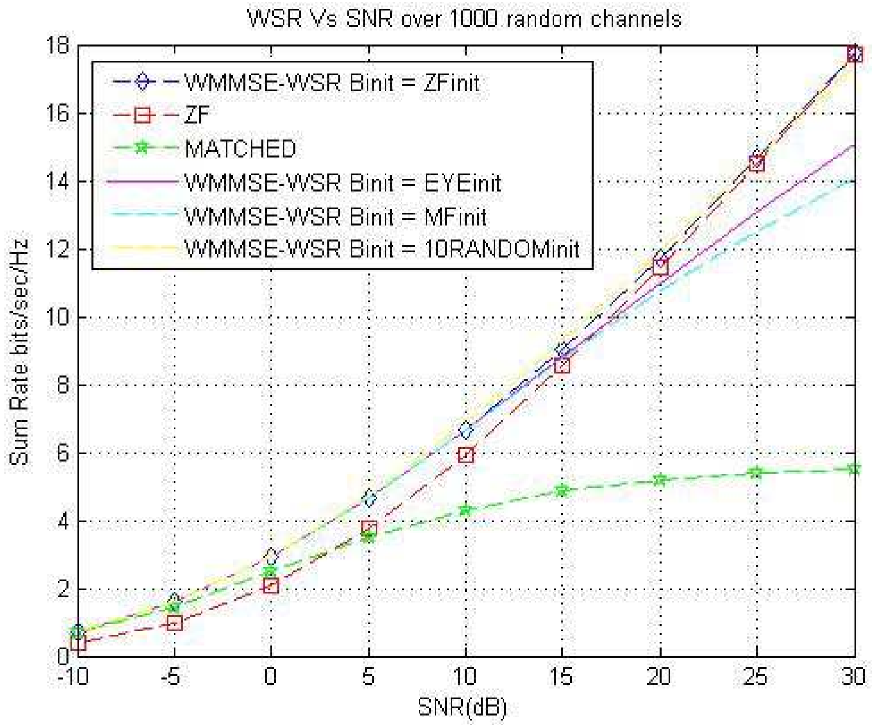


Figure 5.4: Sum-rate performance averaged over 1000 random channels

As it can be seen in figure 5.4 the best performance is given by WMMSE algorithm, specifically WMMSE initialized with transmitted filter Zero-forcing and WMMSE initialized with 10 random transmitted filters. These performances show that depend the initialization of WMMSE it can be found different local optimums. Deeper study about the optimums of the algorithm is shown in [5].

5.2. Rayleigh channels

Real measurement data have been analyzed through a program as it can be seen in annex 1. For the purpose to check the channel processing, it have been tested by entry random coefficients; all of them with Rayleigh fading distribution.

The generation of the Rayleigh samples has been done as:

```
Hki = (1/sqrt(2))*(randn(x,y)+i*randn(x,y))
```

For the purpose to check if channels coefficients follow Rayleigh distribution, some test has been done.

Histogram

Rayleigh distribution has probability density function which shows the probability to get one specific value (Figure 2.2). Histogram means how many times we have values in specific interval, hence histogram is the same like pdf but without normalized.

The histogram of each channel has been done as figure 5.5 shows.

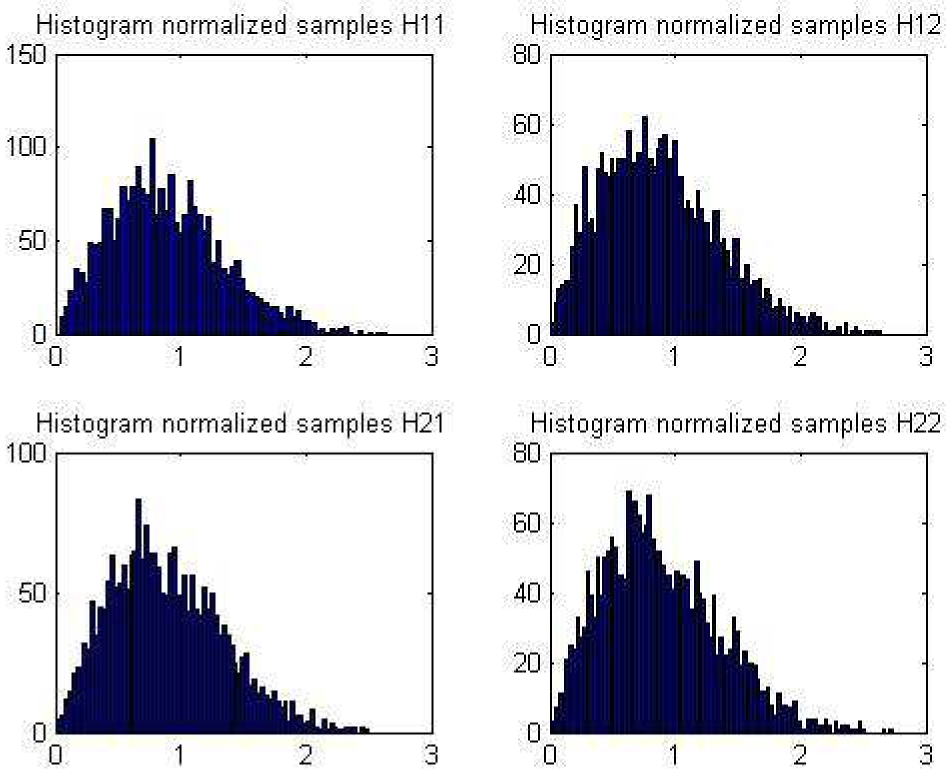


Figure 5.5: Histogram samples for each channel

As the channels have been done with Rayleigh fading distribution, the histogram results are perfectly like pdf Rayleigh distribution.

Cumulative Density Function

CDF is the function which shows the probability of a random sample can have less or equal than a certain value x.

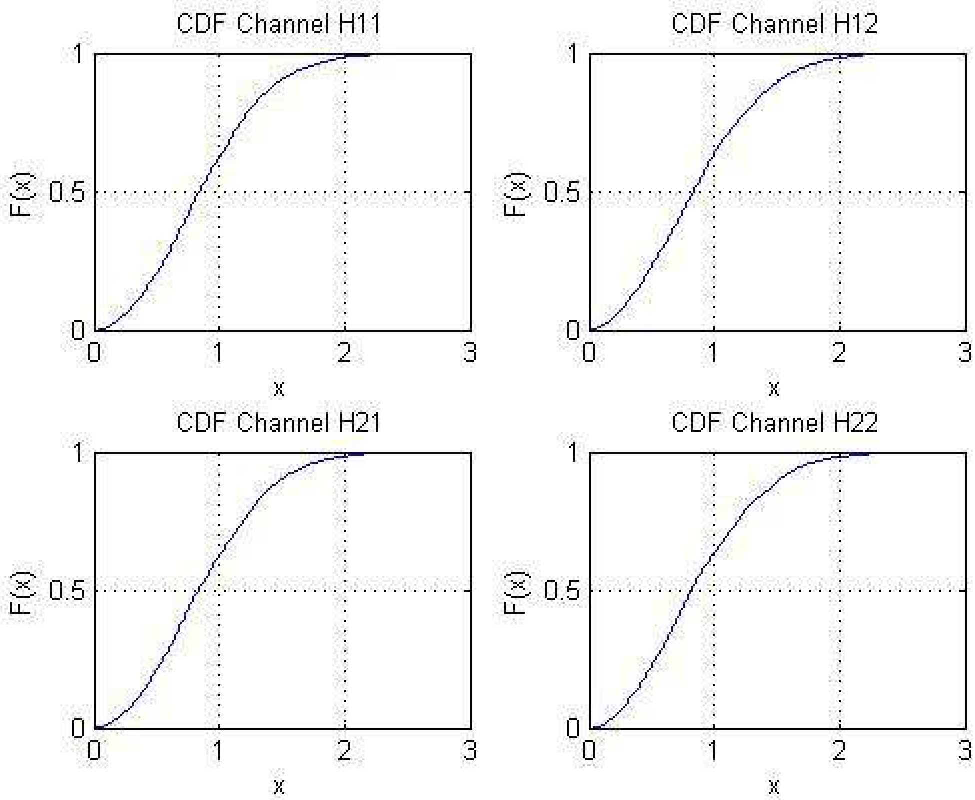


Figure 5.6: CDF of each channel

The four links have the same CDF and it can be seen in figure 5.6 the theoretical CDF for Rayleigh distribution.

Probplot

This Matlab function is used only for verifying the distribution of the samples. It can be seen in figure 5.7 the behavior of the samples follows perfectly Rayleigh distribution.

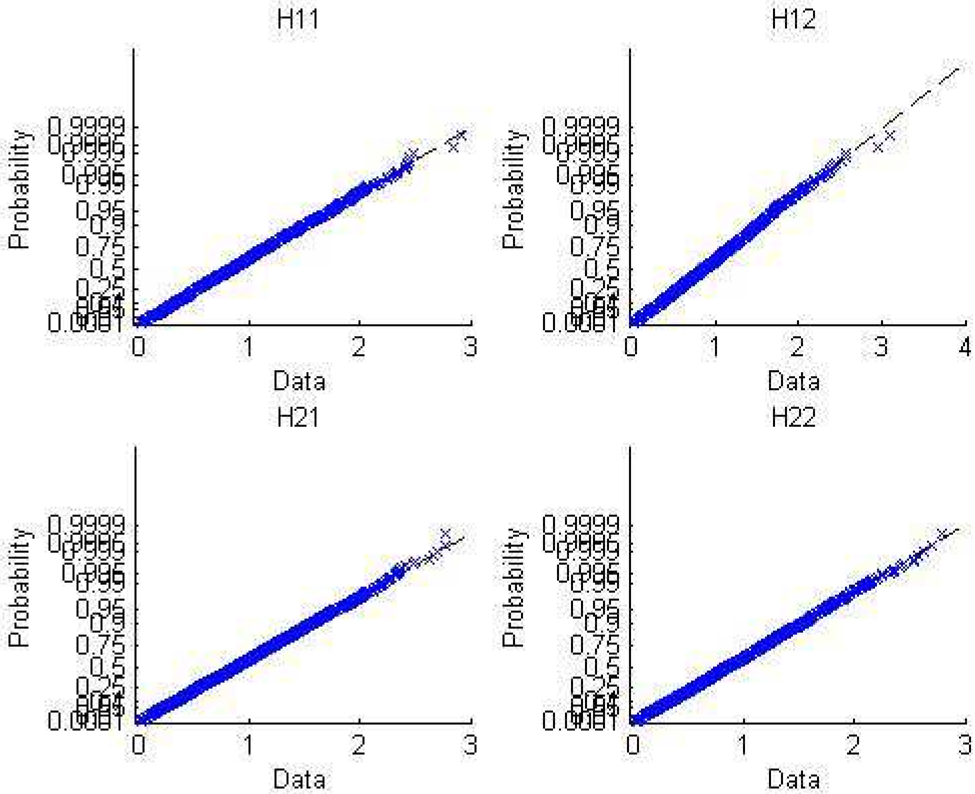


Figure 5.7: Probplot of each channel

Cross-correlation and autocorrelation

As it have been seen in figure 4.10 and 4.11 the cross-correlation between two randomized samples is always 0. On the other hand, autocorrelation is always 0, but not in 0 delay (lag), where the value is 1. Results are shown in figure 5.8, 5.9 and 5.10.

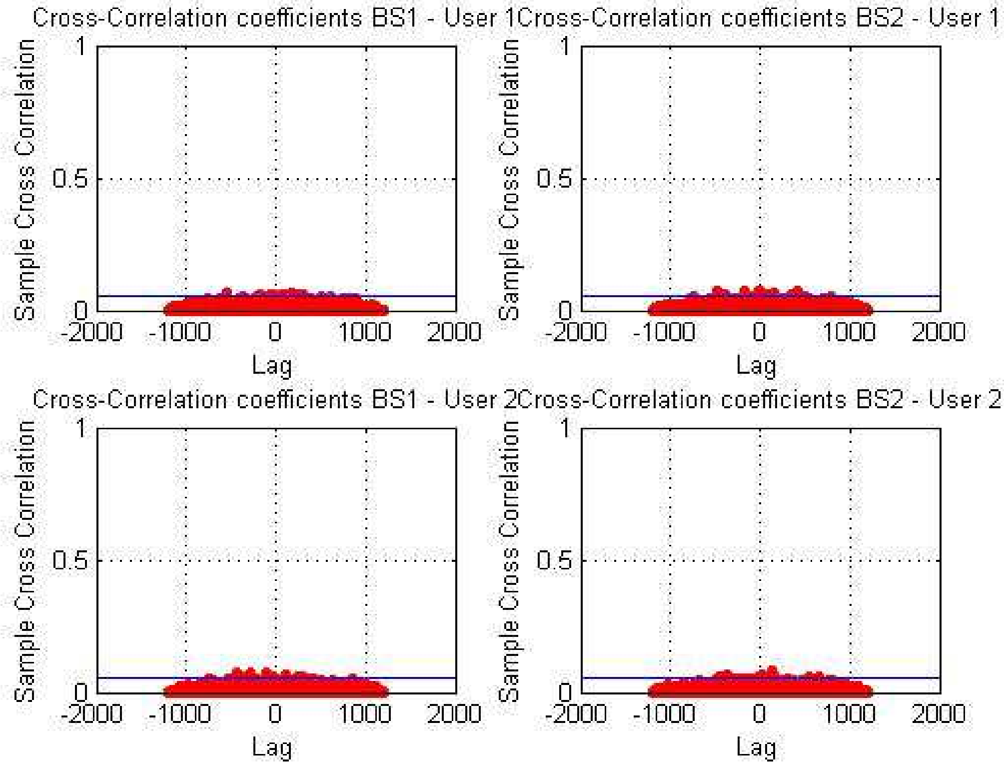


Figure 5.8: Cross-correlation between transmit antennas working at same base station.

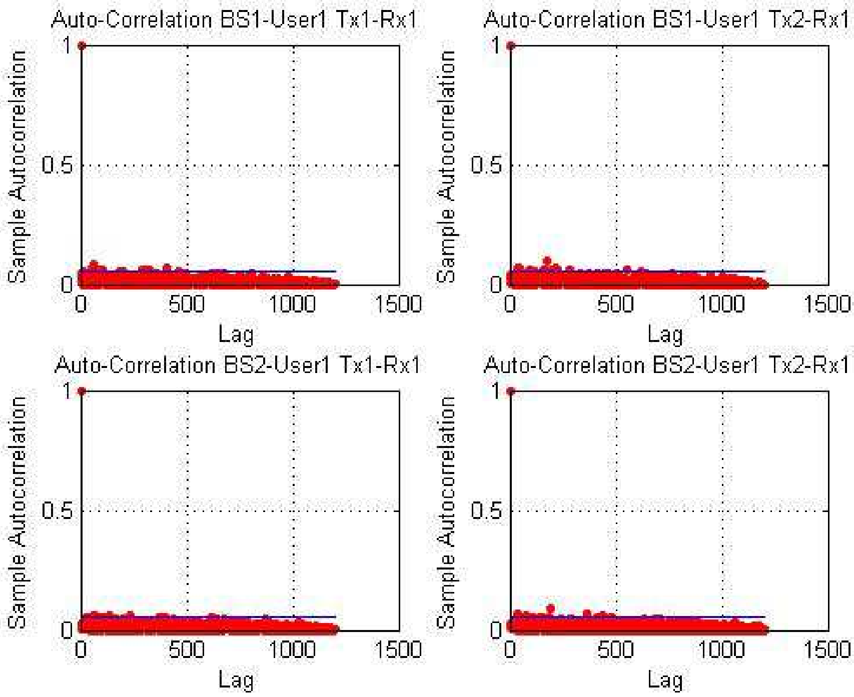


Figure 5.9: Auto correlation between each link for user 1

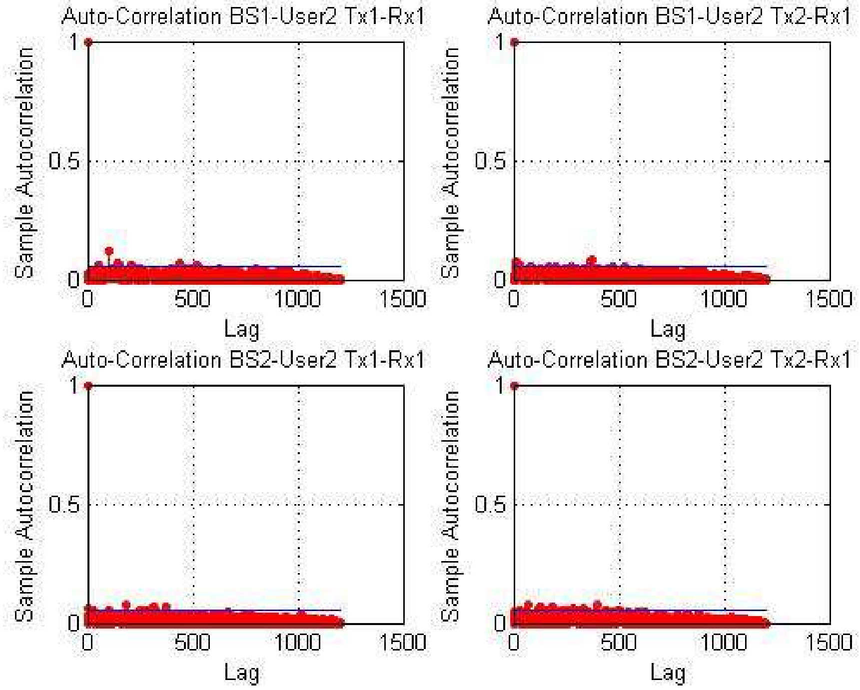


Figure 5.10: Auto correlation between each link for user 2

Comparison with theoretical channel

Once the study on the data has been done; it is time to compare the performance of the channel through the algorithm and compare final performance with the theoretical results.

In that case, theoretical result and data analyzed should be almost the same since the values for the channels have been created by the same way. As it can be seen in figure 5.11 the results have the same trend as it has been expected.

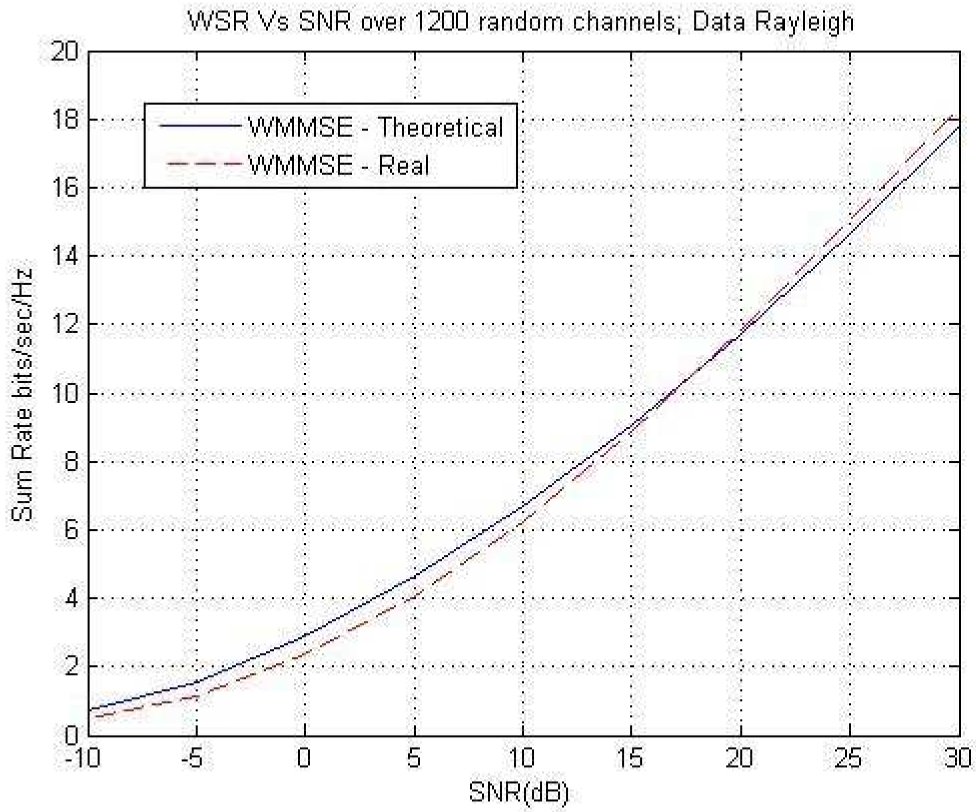


Figure 5.11: Sum-rate performance averaged 1200 channels with Rayleigh channels

5.3. Real measurement data 1

The same analysis on real data has been done as Rayleigh channels and the results have been compared. Also the power delay profile is shown as well as the gain of the channels.

5.3.1. Reference handset normalization

The channel between base station 1 and user 1, $\boldsymbol{H}_{11}$, has been taken as a reference.

Power Delay Profile

As it is mentioned at chapter 4, 1200 positions have been taken during the data measurements. Power Delay Profile shows us, in a concretely position, how much power the handset has received from the base station. The power does not arrive at the same time; it has a certain delay which it can be seen in abscissa axis in figure 5.12.

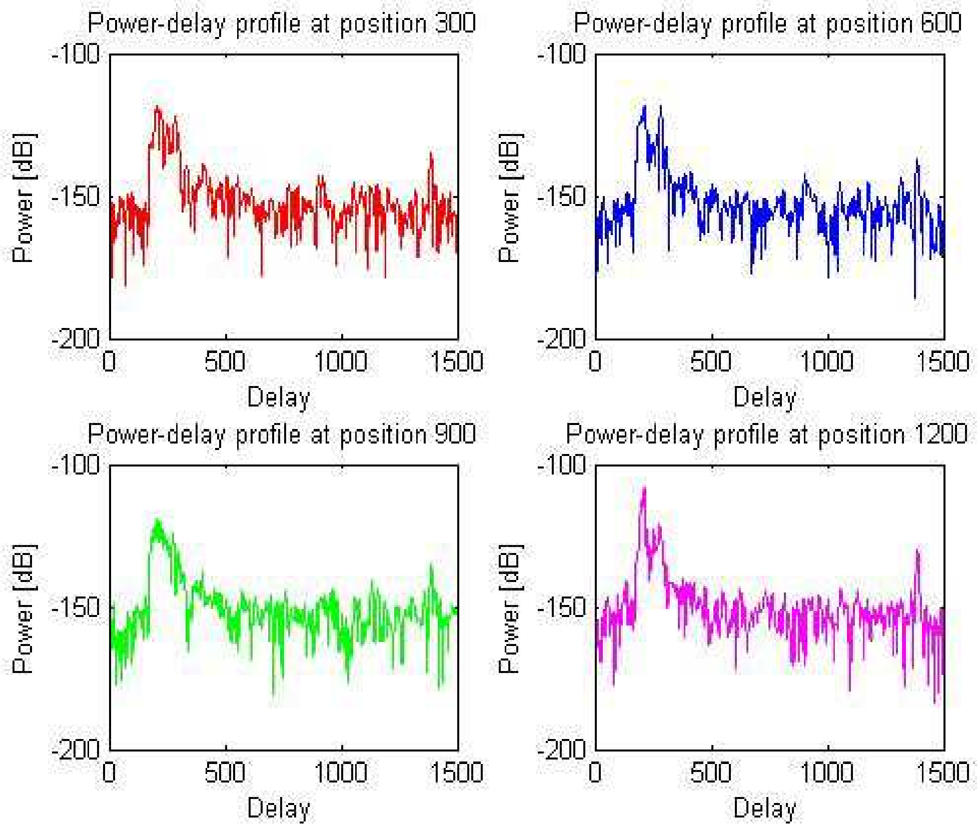


Figure 5.12: Power Delay Profile data 1 measured in four different positions: 300, 600, 900 and 1200, respectively

In real measurement data is supposed to have Rayleigh fading distribution. However, depends the obstacles or path propagation, transmitted signal also can seem more as Rician fading distribution. As it can be seen at figure 5.13 all histograms follow Rayleigh distributions with different mean and variances. To be sure that we are treating with Rayleigh or Rician fading distributions, CDF and probplot of the channels have also been demonstrated in figures 5.15 and 5.16.

As it can be seen in the next figure, histograms are closer to theoretical histograms shown in 5.5. For the purpose to verify that the real measurement follows Rayleigh distribution, the statistics for each channel has been calculate and simulated with theoretical Rayleigh samples as it can be seen in figure 5.14, where the histograms have been normalized and shown as a pdf. It cannot be forgotten that in real measurements many factors can change the distribution of the samples, for example slow fading, fast fading, propagation path and so on. That is because real measurement data is closer to Rayleigh distribution instead to be equal.

Histogram

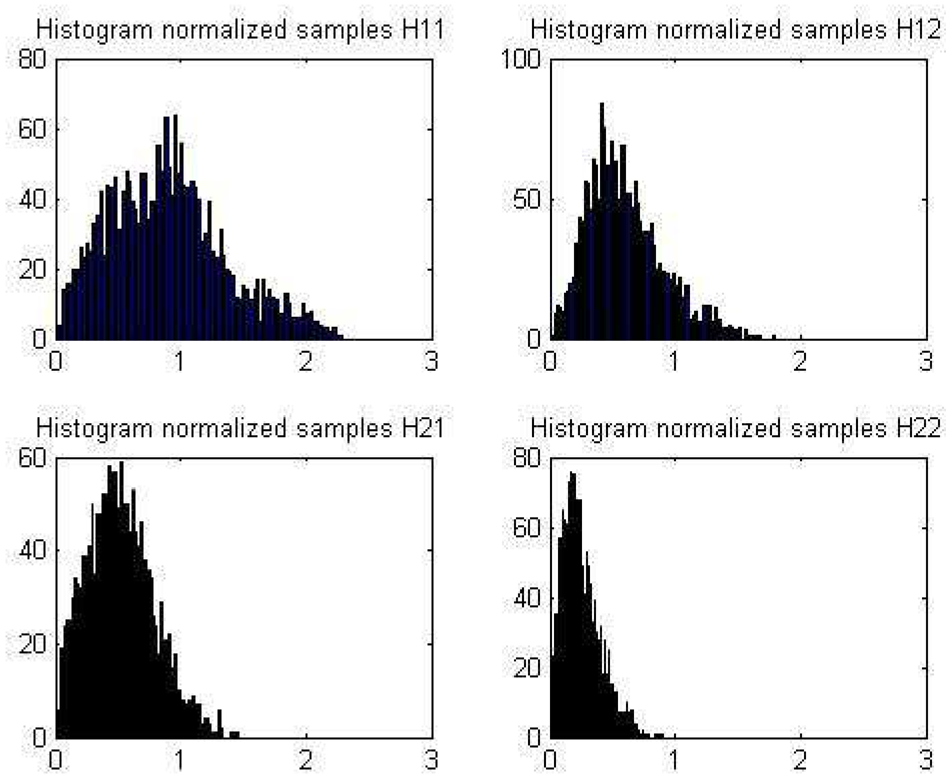


Figure 5.13: Histograms for data 1 per each channel

Histogram normalized

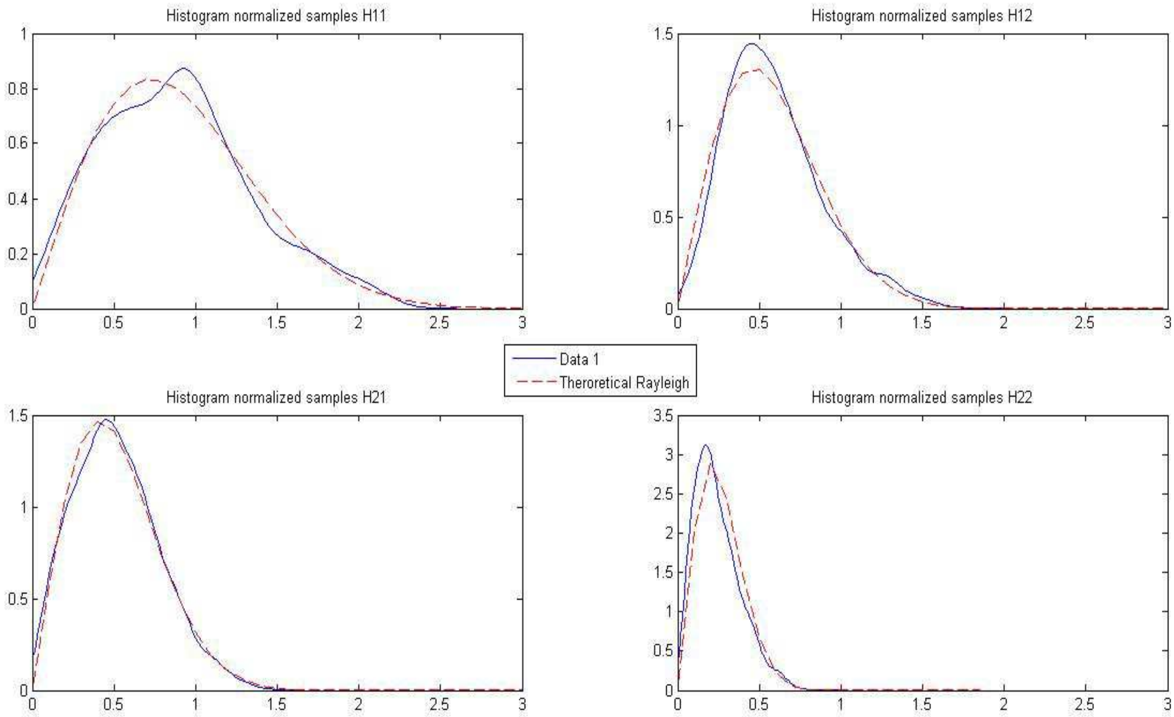


Figure 5.14: Normalized histogram (PDF) data 1 per each channel and comparison with theoretical

CDF

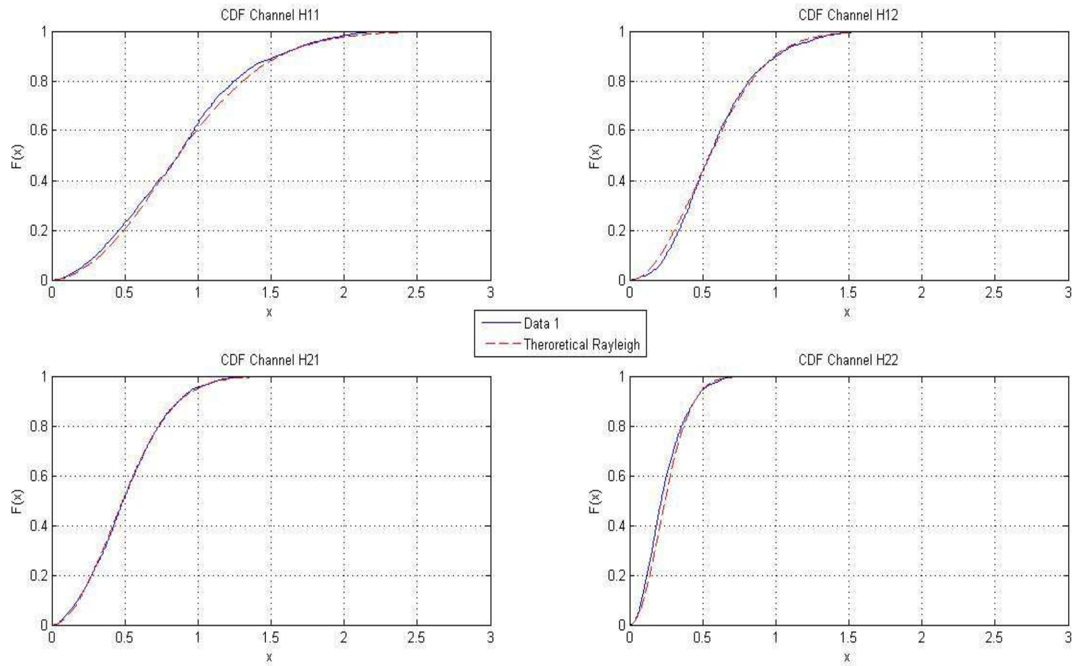


Figure 5.15: Cumulative density function for data 1 per each channel and comparison with theoretical

Probplot

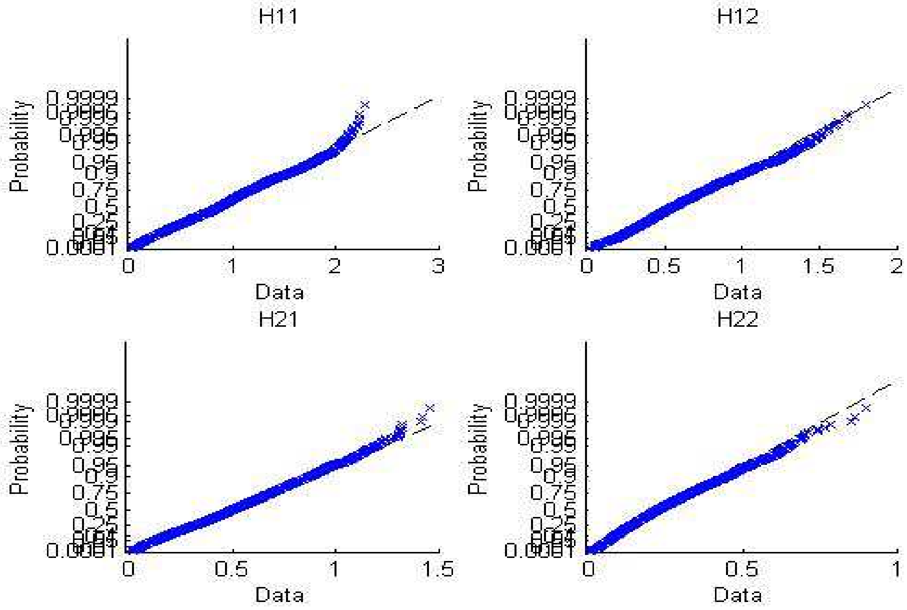


Figure 5.16: Probplot for data 1 per each channel

Gains for each coefficient are shown in figures 5.17 and 5.18. They have been calculated after the channel normalization and it illustrates the difference in gains between each transmit antenna. The analysis has been done per each coefficient and separately for each user.

Gain

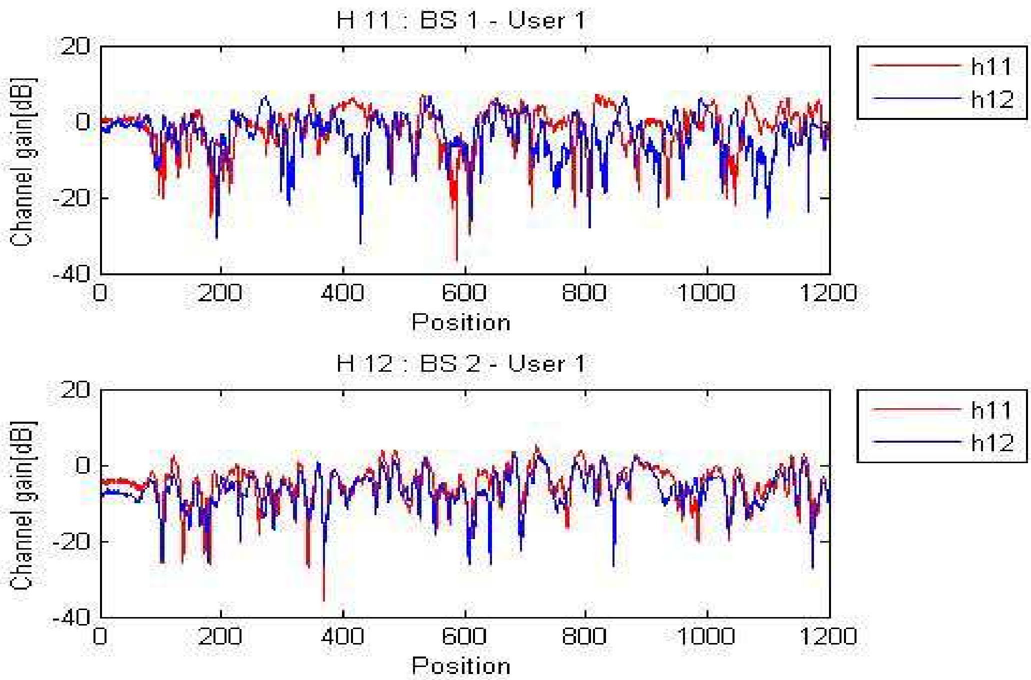


Figure 5.17: Normalized reference handset gains for data 1. Measured in user 1

As it can be seen in figure 5.17 receive gain for base station 1 (the closer one) is high than base station 2, the far base station. This is the expected result, but the gain not only depends how far away is the base station, also depends the way we are keeping the handset (Figure 4.3) and conditions in path propagation at time of the measurements.

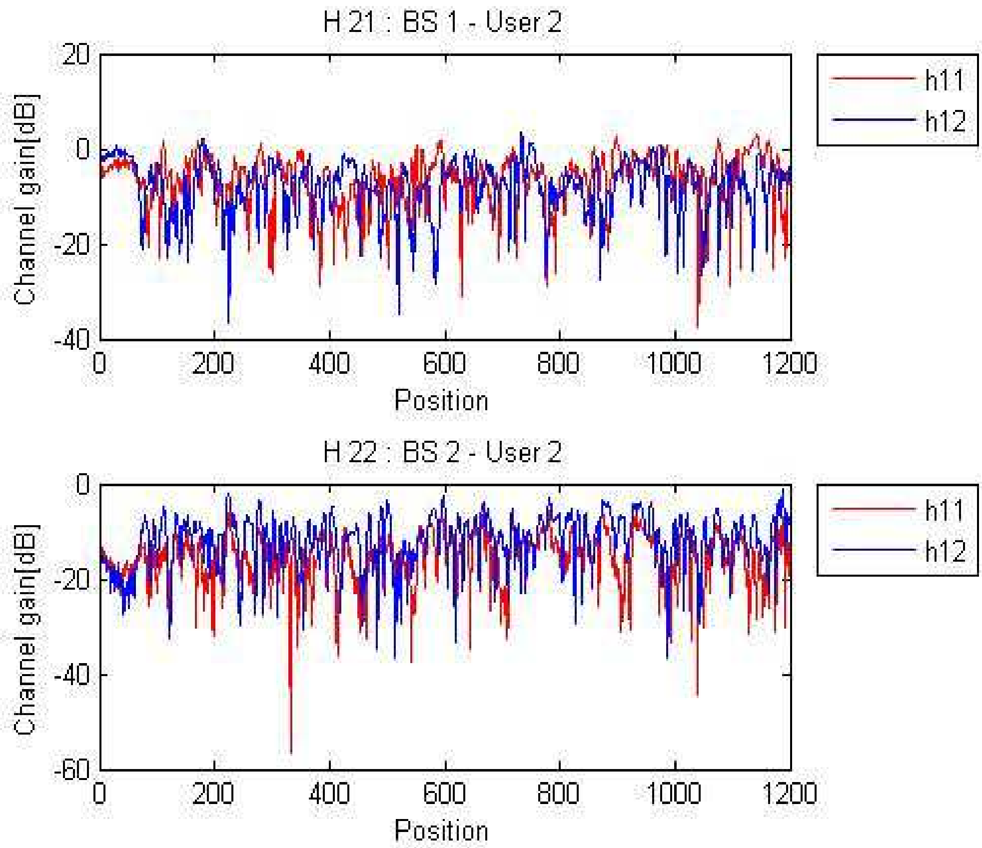


Figure 5.18: Normalized reference handset gains for data 1. Measured in user 2

It is not expected to have perfect uncorrelated coefficients because we do not have perfect Rayleigh or Rician samples as it can be seen in above figures, it means that the received data are not totally randomized, so we expect some kind of correlation between coefficients. Moreover, as it is mentioned in chapter 4, one base station has 60 meters of height; it means it has more directly path propagation with the handsets, at least much more than the base station at lower position. It implies that the signal which is received does not have much fast fading, in other words, the received signal is clearer. This effect can be seen in the plots of the gain (5.17 and 5.18), where the channels H12 and H22 have a softer trend in comparison with the others.

This property of base station 2 has directly effect on cross-correlation between its own coefficients. As the signal belongs to base station 2 does not have much fast fading, both follows almost the same trace during the positions. It is hard to see in figures 5.17 and 5.18, but as it is shown in figure 5.19 cross correlation shows that the coefficients in base station 2 have a high similarity between them.

Cross-correlation and autocorrelation

Auto correlation is shown in figures 5.20 and 5.21 for the purpose to verify if the coefficients of the channels are random.

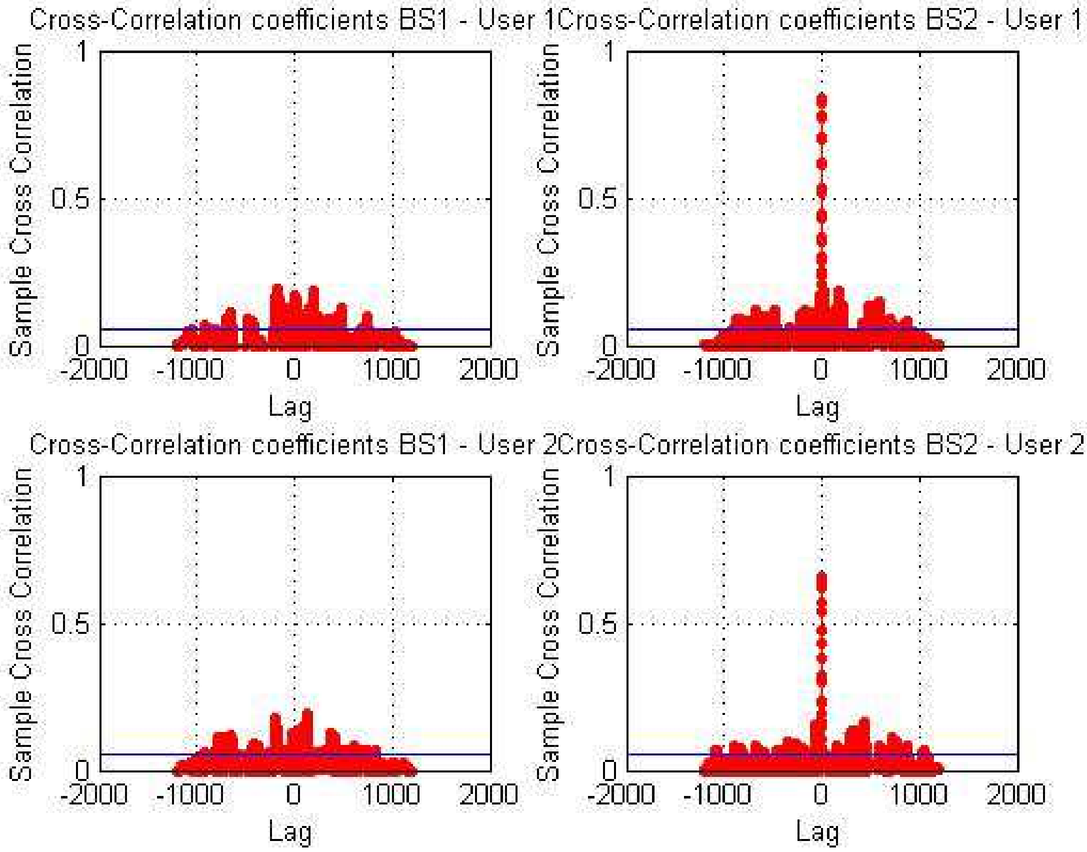


Figure 5.19: Cross correlation for data 1. It is shown per each channel.

Auto correlation have been computed per each coefficient, separately them depending the user they serve. In figure 5.20 it is shown auto correlations for each coefficients which to user 1. In figure 5.21 refers to user 2.

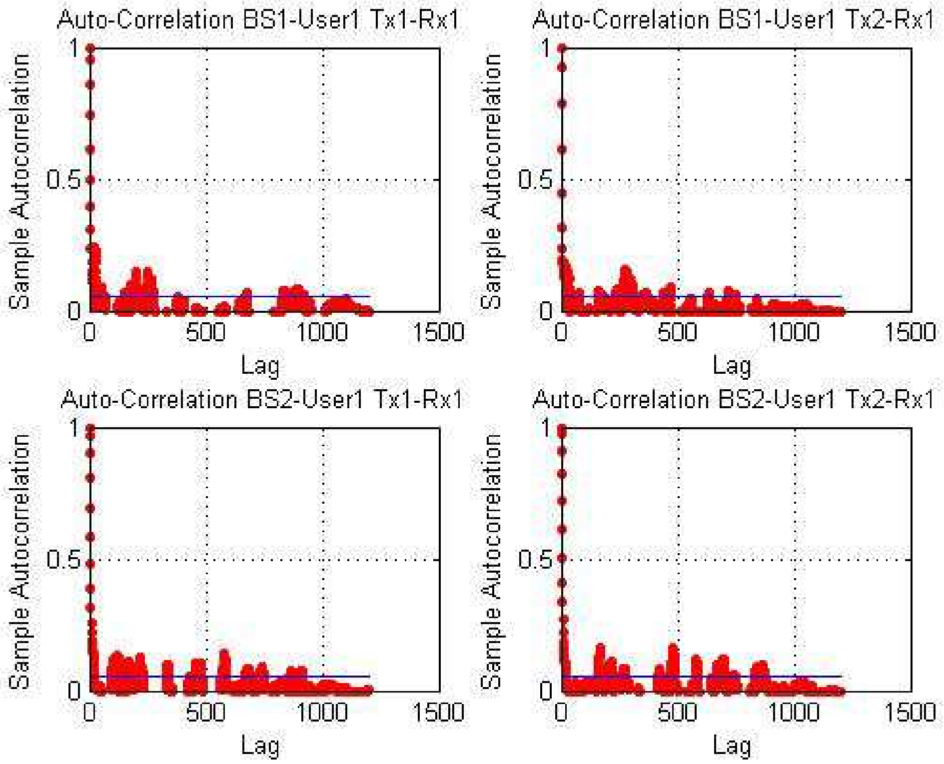


Figure 5.20: Auto correlation for data 1. Computed each coefficient for user 1

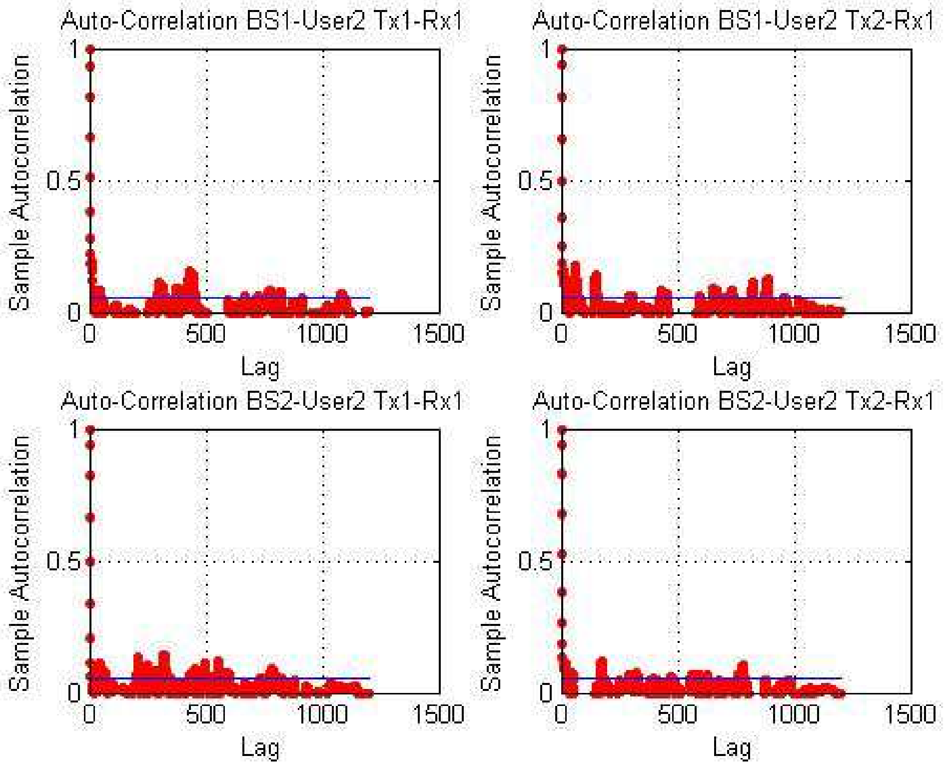


Figure 5.21: Auto correlation for data 1. Computed each coefficient for user 2

Comparison with theoretical channel

It is time to compare the performance of the channel through the algorithm and compare final performance with the theoretical results as it can see in figure 5.22.

It can be seen the numerical results in Table 5.1.

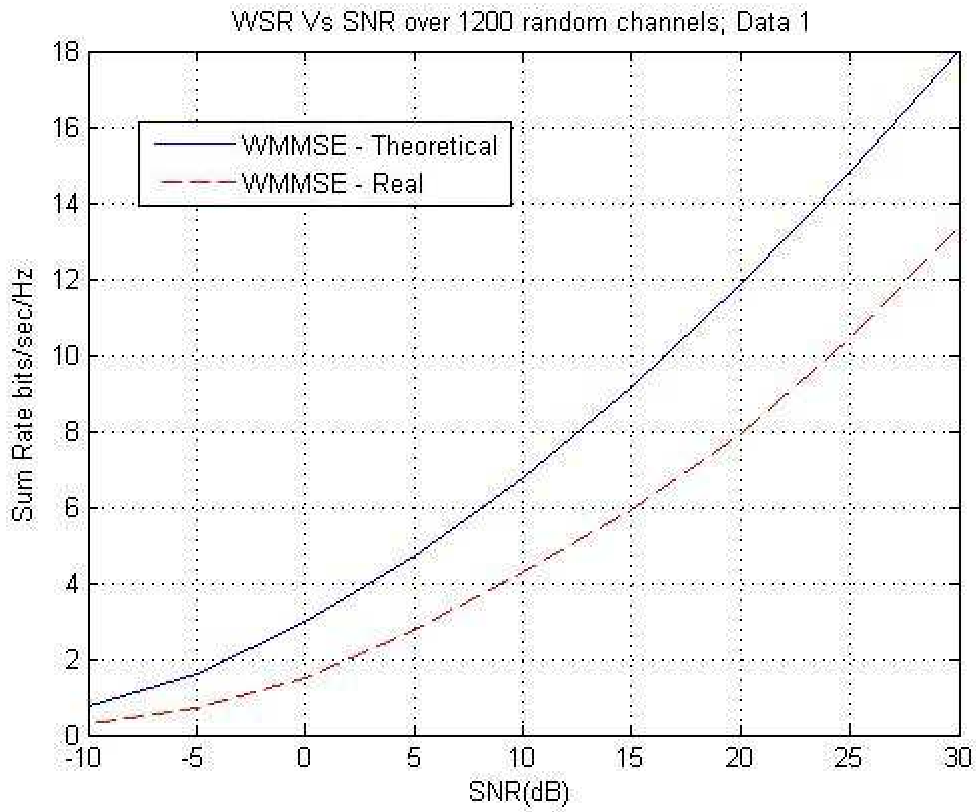


Figure 5.22: Sum-Rate for data 1 using reference handset normalization

		SNR (dB)								
	Ref.Handset	-10	-5	0	5	10	15	20	25	30
Sum - Rate	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77
Bits/sec/Hz	Real Data_1	0,27	0,69	1,51	2,74	4,28	5,91	7,90	10,44	13,39

Table 5.1: Overview Sum-Rate for data 1 using reference handset normalization

5.3.2. Each link normalization

This normalization has been done for the purpose to skip the effect of the branch power ratio between links. All links have the same gain, however the rest of properties (correlations, PDF, CDF) are kept. As it can be seen in figure 5.23 and 5.24 the gains do not have a lot of difference between them like in figures 5.17 and 5.18.

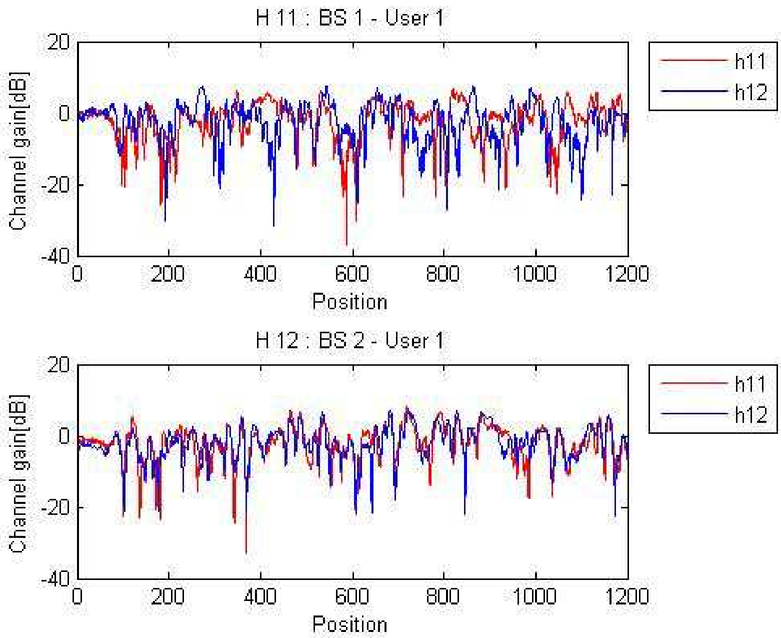


Figure 5.23: Normalized each link gains for data 1. Measured in user 1

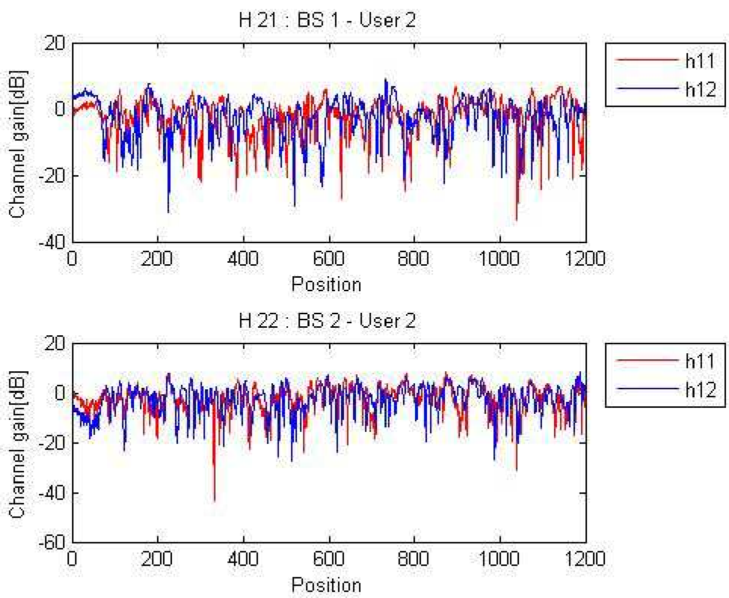


Figure 5.24: Normalized each link gains for data 1. Measured in user 2

Comparison with theoretical channel

As it can see in figure 5.25 the performance if branch power is skipped is high.

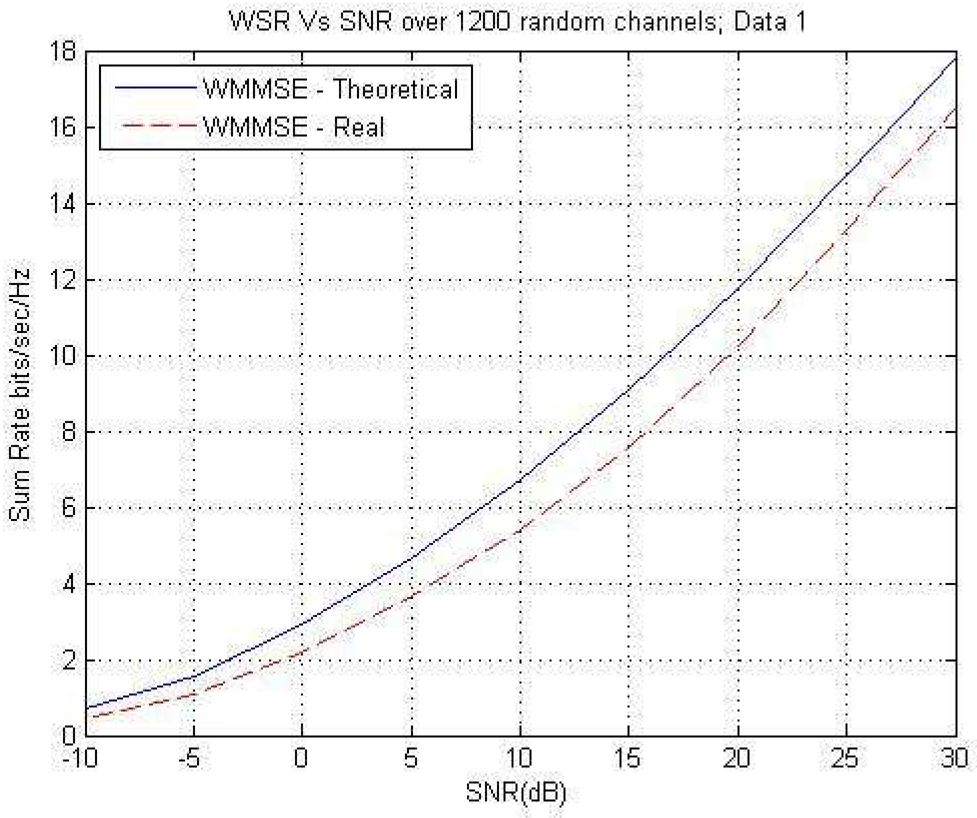


Figure 5.25: Sum-Rate for data 1 using each link normalization

		SNR (dB)								
	Each Link	-10	-5	0	5	10	15	20	25	30
Sum - Rate	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77
Bits/sec/Hz	Real Data_1	0,42	1,09	2,17	3,63	5,40	7,50	10,24	13,29	16,50

Table 5.2: Overview Sum-Rate for data 1 using each link normalization

5.3.3. Removing cross correlation effect

As it is commented in 4.2.1.3 cross correlation effect are removing changing the positions of the channels for base station 2. The gains and the rest of properties (auto correlation, PDF, CDF) are the same like in reference handset normalization.

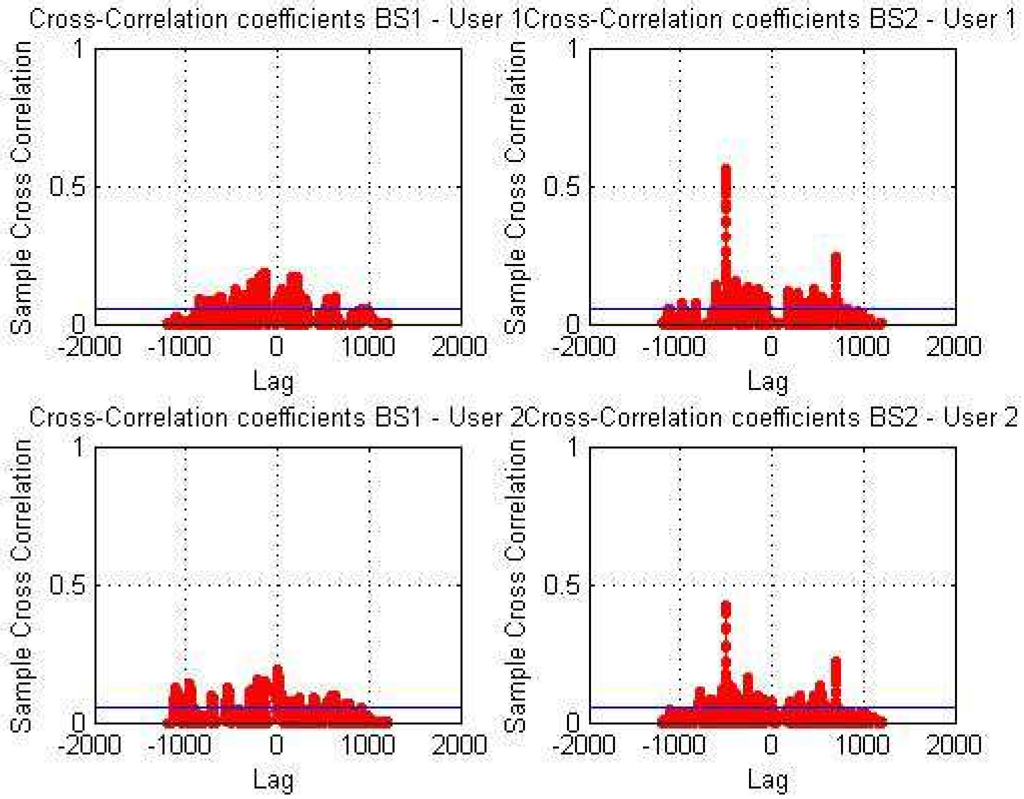


Figure 5.26: Cross correlation removing for data 1. It is shown per each channel.

Comparison with theoretical channel

As it can see in figure 5.27 the performance if cross correlation is removed got better in comparison with figure 5.22, where cross correlation and gain effects are in.

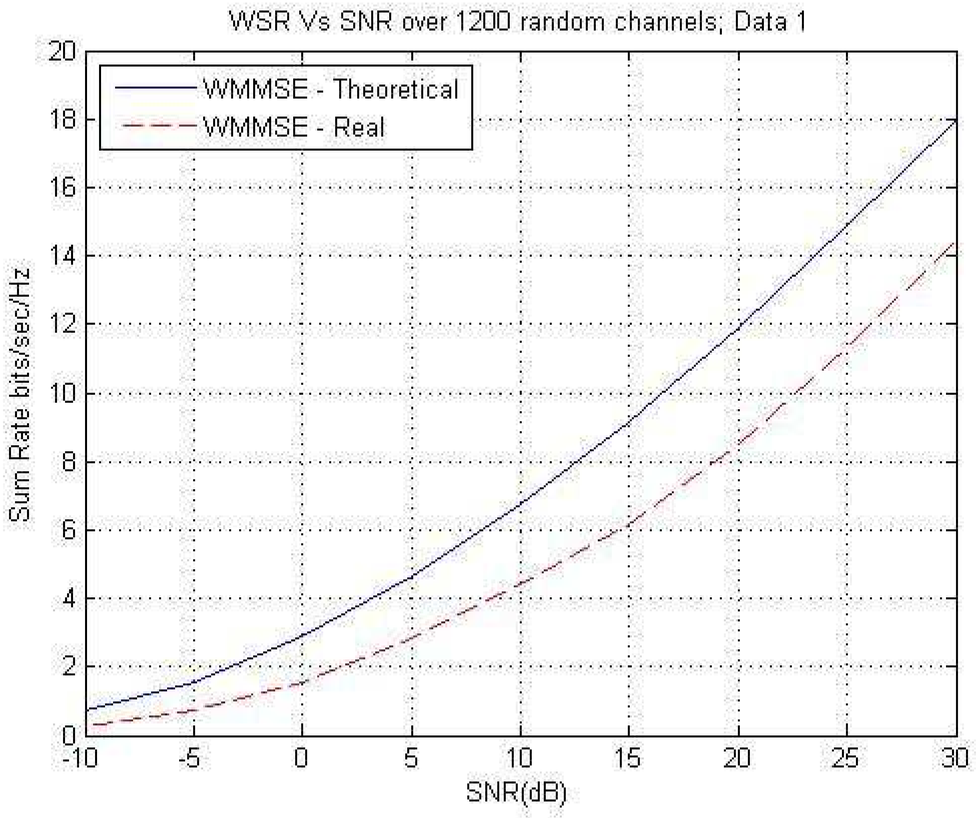


Figure 5.27: Sum-Rate for data 1 removing cross correlation

		SNR (dB)								
	Removing Cross	-10	-5	0	5	10	15	20	25	30
Sum - Rate	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77
Bits/sec/Hz	Real Data_1	0,27	0,70	1,53	2,80	4,40	6,18	8,48	11,33	14,44

Table 5.3: Overview Sum-Rate for data 1 removing cross correlation

5.3.4. Overview real measurement data 1

As it can be seen in figure 5.28 if all features of the channel are kept with reference handset normalization the worst performance of sum-rate is achieved. On the other hand, sum-rate has had an improvement when each link normalization and removing cross correlation has been done. Each link normalization allows skipping the effect of branch power ratio as it can see in section 5.3.2 and the best sum-rate is achieved with this normalization. In the method for removing cross correlation, branch power ratio is conserved, but the effect of cross correlation is removed. It lets see how much cross correlation degrades the sum-rate.

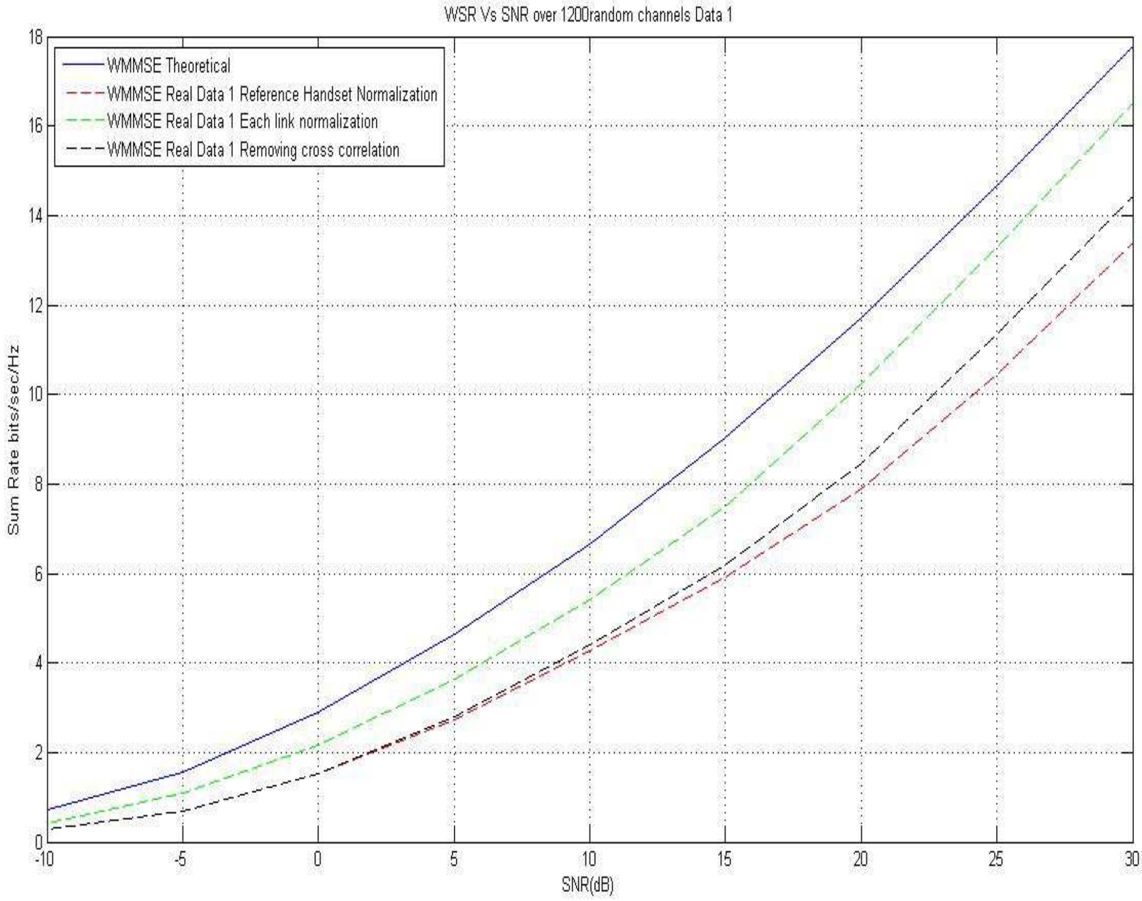


Figure 5.28: Overview Sum-rate data 1 for all normalizations

As it can be seen in Table 5.4 sum up of all different normalizations have been done.

The results show that the effect of the branch power is more prejudicial than the effect of the cross correlation when sum rate us computed.

Around 1.5 bits/sec/Hz is lost when the effect of branch power is. Loss due to cross correlation is less than the branch power and it is around 0.4 bits/sec/Hz

			SNR (dB)									Difference
	Normalization		-10	-5	0	5	10	15	20	25	30	Average
Sum - Rate	Reference	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77	
Bits/sec/Hz	Handset	Real Data_1	0,27	0,69	1,51	2,74	4,28	5,91	7,90	10,44	13,39	
Difference	Theoretical	Real Data_1	0,44	0,88	1,39	1,89	2,39	3,12	3,83	4,23	4,38	2,51
Sum - Rate	Each	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77	
Bits/sec/Hz	Link	Real Data_1	0,42	1,09	2,17	3,63	5,40	7,50	10,24	13,29	16,50	
Difference	Theoretical	Real Data_1	0,29	0,48	0,73	1,00	1,27	1,53	1,49	1,38	1,27	1,05
Sum - Rate	Removing	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77	
Bits/sec/Hz	cross-corr	Real Data_1	0,27	0,70	1,53	2,80	4,40	6,18	8,48	11,33	14,44	
Difference	Theoretical	Real Data_1	0,44	0,87	1,37	1,83	2,27	2,85	3,25	3,34	3,33	2,17

Table 5.4: Overview Sum-rate data 1 for all normalizations

5.4. Real measurement data 2

For the purpose to verify the results obtain with data 1, the same procedure has been done for two more real measurement data. As the procedure has been the same like in data 1, only results are shown for data 2 and data 3.

5.4.1. Reference handset normalization

Comparison with theoretical channel

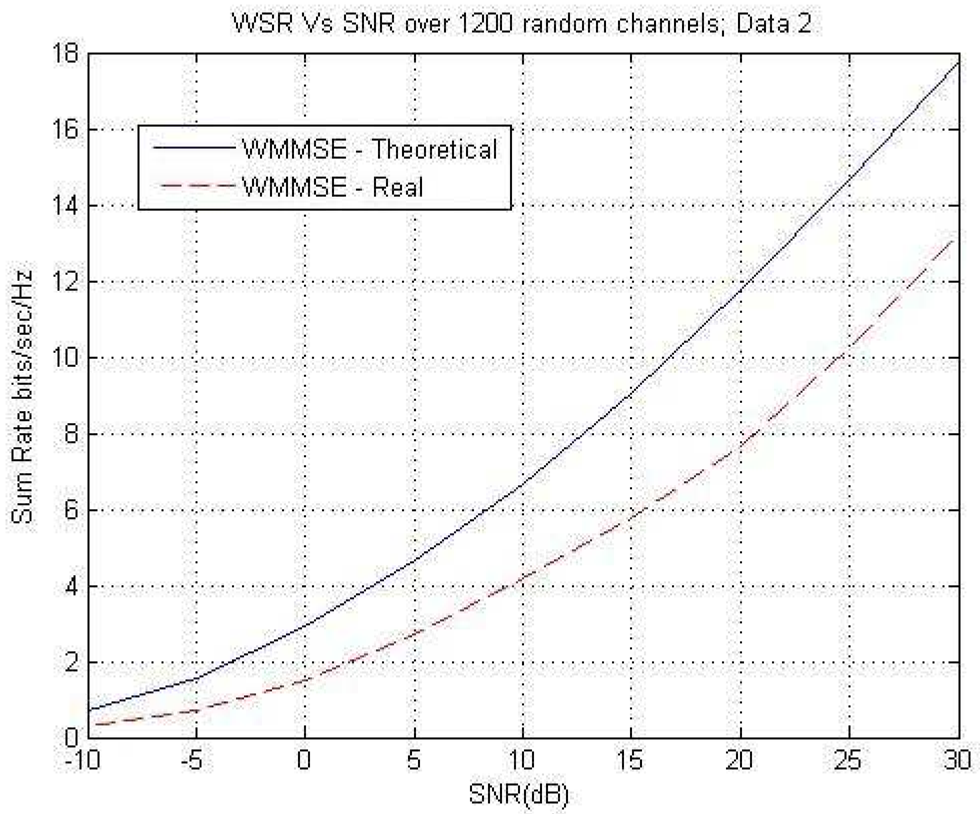


Figure 5.29: Sum-Rate for data 2 using reference handset normalization

		SNR (dB)								
	Ref.Handset	-10	-5	0	5	10	15	20	25	30
Sum - Rate	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77
Bits/sec/Hz	Real Data_2	0,26	0,68	1,48	2,68	4,17	5,75	7,66	10,26	13,25

Table 5.5: Overview Sum-Rate for data 2 using reference handset normalization

5.4.2. Each link normalization

Comparison with theoretical channel

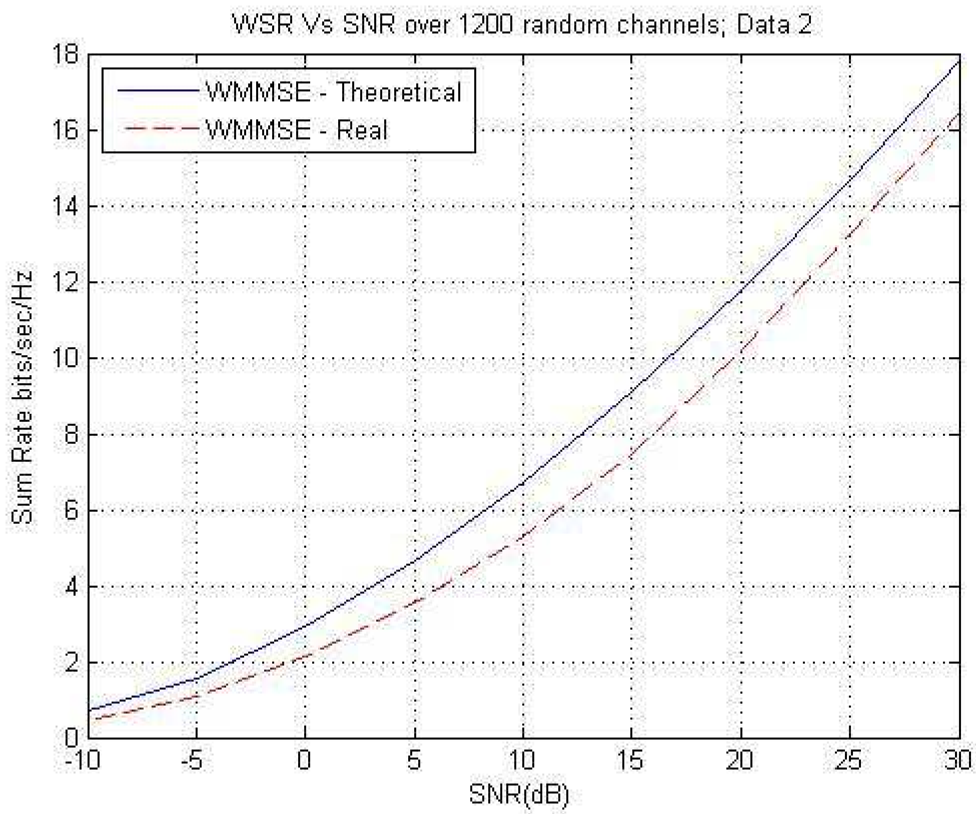


Figure 5.30: Sum-Rate for data 2 using each link normalization

		SNR (dB)								
	Each Link	-10	-5	0	5	10	15	20	25	30
Sum - Rate	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77
Bits/sec/Hz	Real Data_2	0,45	1,08	2,13	3,57	5,31	7,45	10,16	13,21	16,42

Table 5.6: Overview Sum-Rate for data 2 using each link normalization

5.4.3. Removing cross correlation

Comparison with theoretical channel

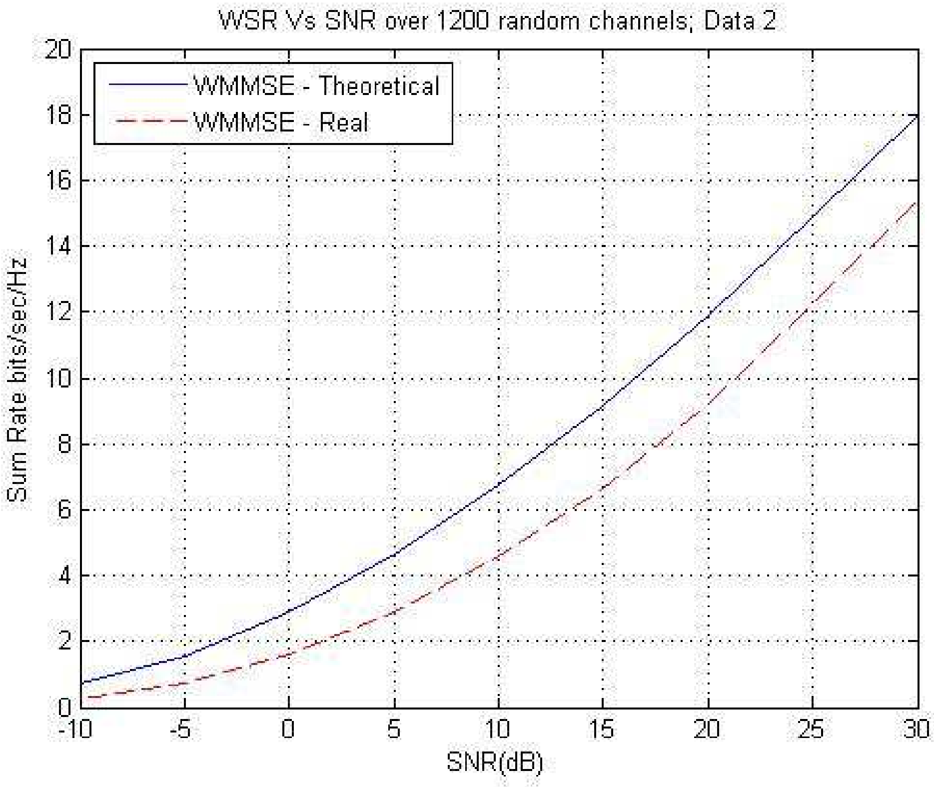


Figure 5.31: Sum-Rate for data 2 removing cross correlation

		SNR (dB)								
	Removing Cross	-10	-5	0	5	10	15	20	25	30
Sum - Rate	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77
Bits/sec/Hz	Real Data_2	0,28	0,73	1,59	2,90	4,58	6,61	9,22	12,23	15,42

Table 5.7: Overview Sum-Rate for data 2 removing cross correlation

5.4.4. Overview real measurement data 2

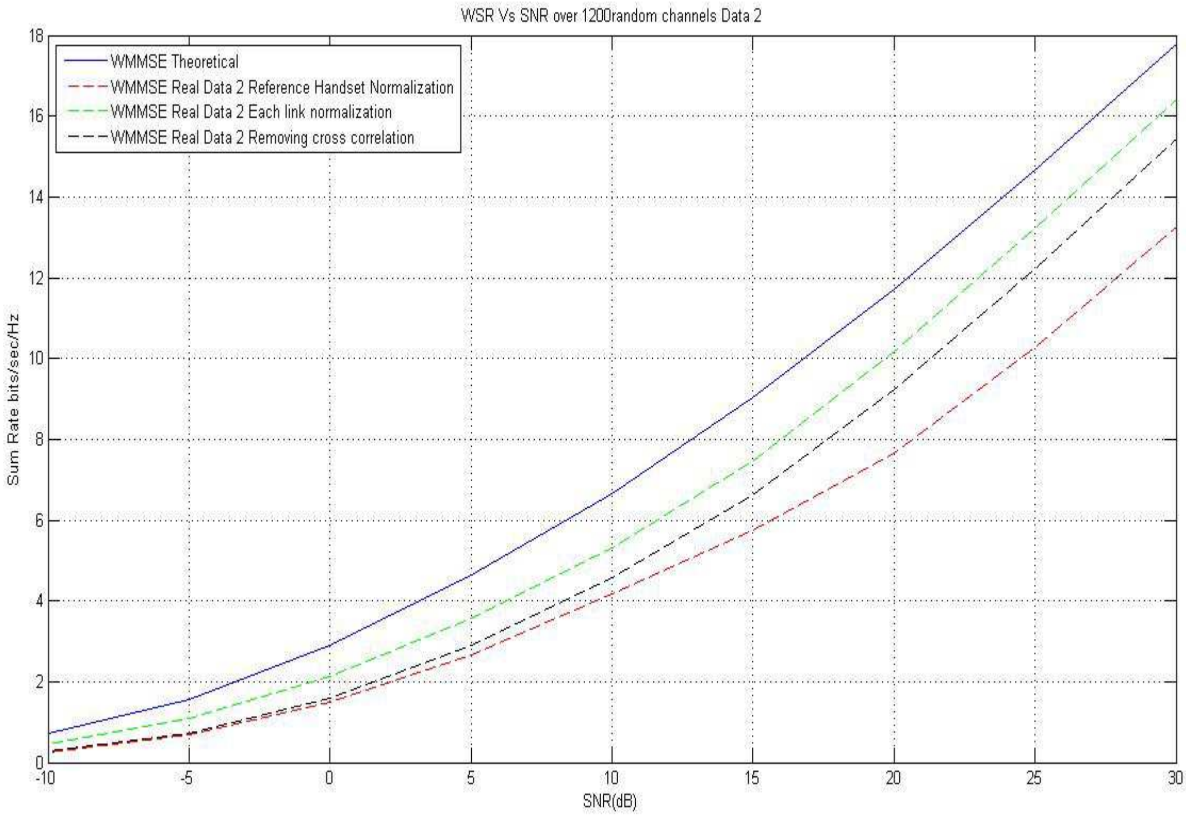


Figure 5.32: Overview Sum-rate data 2 for all normalizations

As it can be seen in table 5.8 the lost due to branch power is around 1.5 Bits/sec/Hz and for the cross correlation is 0.8 Bits/sec/Hz.

	Normalization		SNR (dB) -10	-5	0	5	10	15	20	25	30	Difference Average
Sum - Rate Bits/sec/Hz	Reference Handset	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77	
		Real Data_2	0,26	0,68	1,48	2,68	4,17	5,75	7,66	10,26	13,25	
Difference	Theoretical	Real Data_2	0,45	0,89	1,42	1,95	2,50	3,28	4,07	4,41	4,52	2,61
Sum - Rate Bits/sec/Hz	Each Link	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77	
		Real Data_2	0,45	1,08	2,13	3,57	5,31	7,45	10,16	13,21	16,42	
Difference	Theoretical	Real Data_2	0,26	0,49	0,77	1,06	1,36	1,58	1,57	1,46	1,35	1,10
Sum - Rate Bits/sec/Hz	Removing cross-corr	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77	
		Real Data_2	0,28	0,73	1,59	2,90	4,58	6,61	9,22	12,23	15,42	
Difference	Theoretical	Real Data_2	0,43	0,84	1,31	1,73	2,09	2,42	2,51	2,44	2,35	1,79

Table 5.8: Overview Sum-rate data 2 for all normalizations

5.5. Real measurement data 3

Results for data 3 are shown in this section.

5.5.1. Reference handset normalization

Comparison with theoretical channel

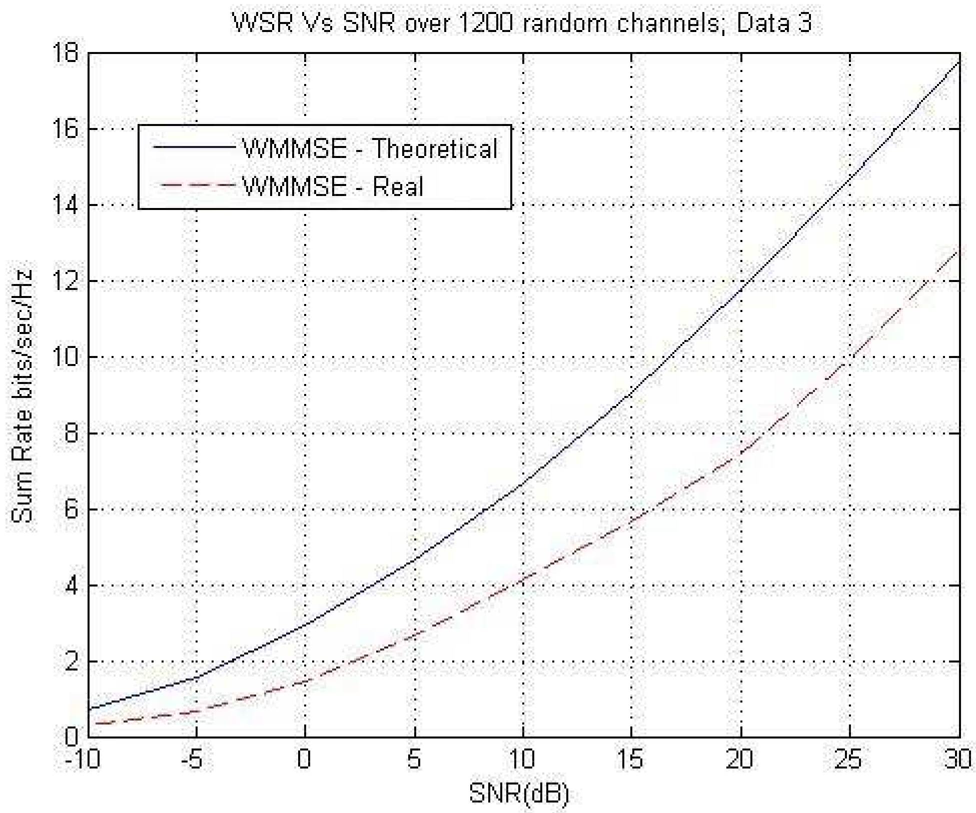


Figure 5.33: Sum-Rate for data 3 using reference handset normalization

		SNR (dB)								
	Ref.Handset	-10	-5	0	5	10	15	20	25	30
Sum - Rate	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77
Bits/sec/Hz	Real Data_3	0,26	0,66	1,45	2,63	4,12	5,67	7,44	9,89	12,79

Table 5.9: Overview Sum-Rate for data 3 using reference handset normalization

5.5.2. Each link normalization

Comparison with theoretical channel

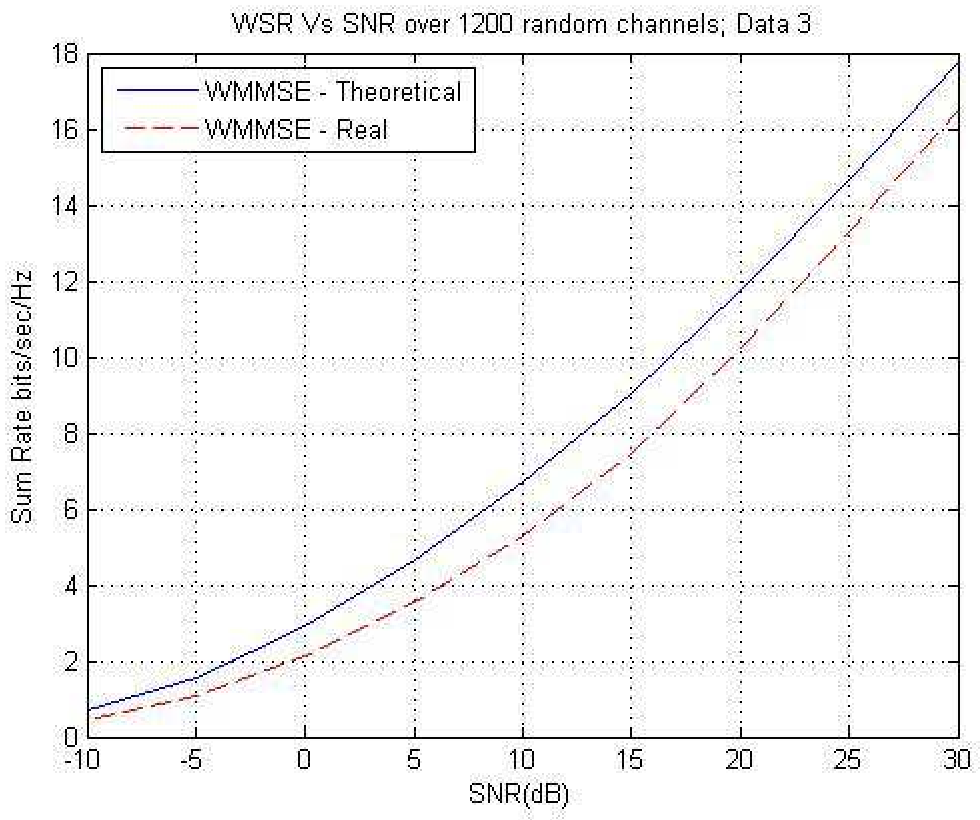


Figure 5.34: Sum-Rate for data 3 using each link normalization

	Each Link	SNR (dB) -10	-5	0	5	10	15	20	25	30
Sum - Rate	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77
Bits/sec/Hz	Real Data_3	0,45	1,07	2,11	3,54	5,26	7,46	10,21	13,27	16,48

Table 5.10: Overview Sum-Rate for data 3 using each link normalization

5.5.3. Removing cross correlation

Comparison with theoretical channel

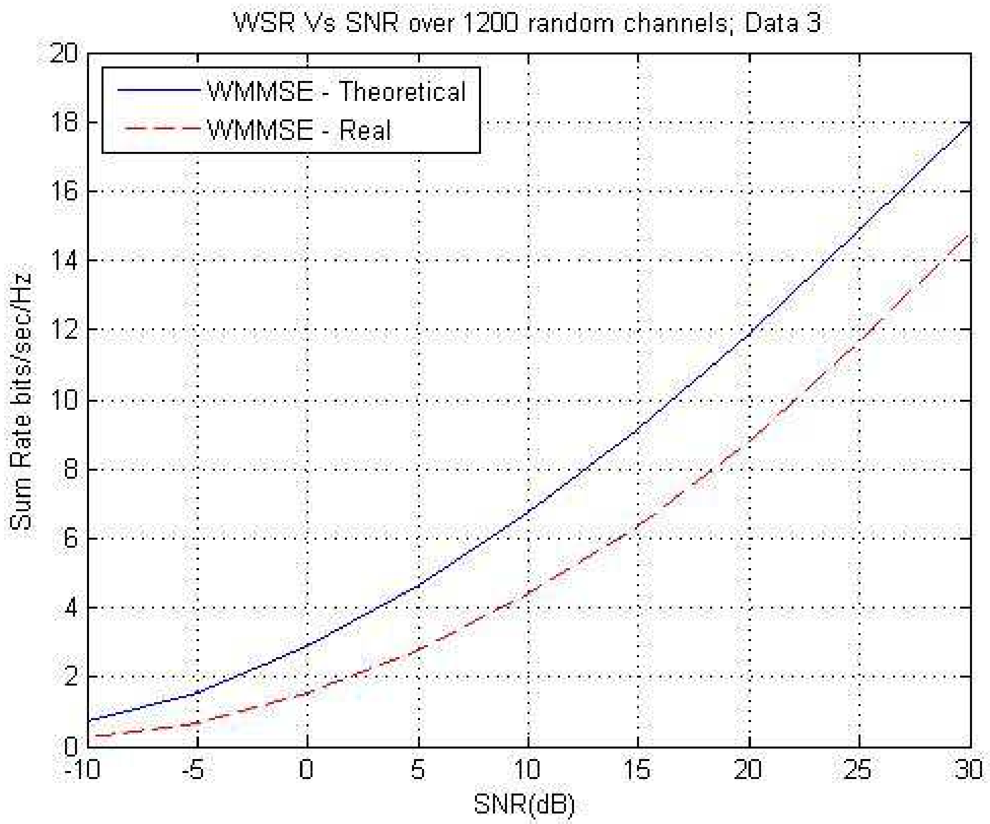


Figure 5.35: Sum-Rate for data 3 removing cross correlation

		SNR (dB)								
	Removing Cross	-10	-5	0	5	10	15	20	25	30
Sum - Rate	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77
Bits/sec/Hz	Real Data_3	0,27	0,69	1,52	2,79	4,42	6,32	8,77	11,69	14,84

Table 5.11: Overview Sum-Rate for data 3 removing cross correlation

5.5.4. Overview real measurement data 3

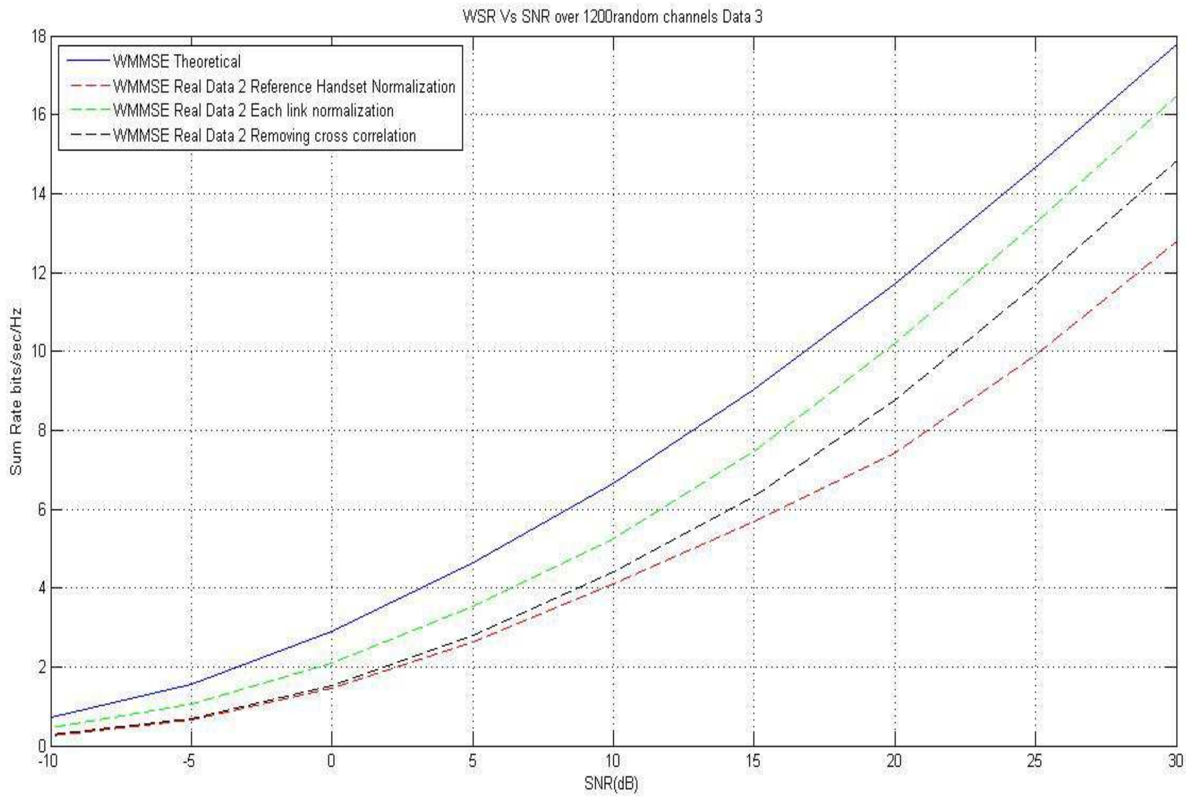


Figure 5.36: Overview Sum-Rate for data 3 all normalizations

As it can be seen in table 5.12 the lost due to branch power is around 1.7 Bits/sec/Hz and for the cross correlation is 0.7 Bits/sec/Hz.

	Normalization		SNR (dB) -10	-5	0	5	10	15	20	25	30	Difference Average
Sum - Rate Bits/sec/Hz	Reference Handset	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77	
		Real Data_3	0,26	0,66	1,45	2,63	4,12	5,67	7,44	9,89	12,79	
Difference	Theoretical	Real Data_3	0,45	0,91	1,45	2,00	2,55	3,36	4,29	4,78	4,98	2,75
Sum - Rate Bits/sec/Hz	Each Link	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77	
		Real Data_3	0,45	1,07	2,11	3,54	5,26	7,46	10,21	13,27	16,48	
Difference	Theoretical	Real Data_3	0,26	0,50	0,79	1,09	1,41	1,57	1,52	1,40	1,29	1,09
Sum - Rate Bits/sec/Hz	Removing cross-corr	Theoretical	0,71	1,57	2,90	4,63	6,67	9,03	11,73	14,67	17,77	
		Real Data_3	0,27	0,69	1,52	2,79	4,42	6,32	8,77	11,69	14,84	
Difference	Theoretical	Real Data_3	0,44	0,88	1,38	1,84	2,25	2,71	2,96	2,98	2,93	2,04

Table 5.12: Overview Sum-rate data 3 for all normalizations

CHAPTER 6

6. CONCLUSIONS

The goal for this project was to find an algorithm for optimizing transmit and receive filter in a cooperative MIMO systems. In a recent paper a relation between WMMSE and WSR was found and it was a motivation to adapt the algorithm in the environment in this project. The cost function of the optimization problem is non-convex; hence we can achieve local optimum solutions. For the purpose to find the best local solution, several initializations for the transmitter have been proposed. The initial transmitter filter which obtains the best performance in sum rate is zero-forcing filter, as well as 10 random initializations filters. It has shown in simulations how the zero-forcing initialization has 4 bits/sec/Hz more than matched-filter initialization and 3 bits/sec/Hz more than identity initialization.

On the other hand, real measurement data was applied to the algorithm to study the effects which it causes in the rate for the users. It has been observed the high correlation between the antennas in base station 2, placed in Aalborg hospital at 60 meters high. The high correlation is due to elevation of this base station: the propagation path to the user is clear, in other words, there are not many obstacles between base station and users. The high correlation implies less randomized samples, which also implies fewer rates for the users. Branch power ratio has been studied as well. For the purpose to check how it degrades the rate, three different normalizations have been done. As it is shown in the simulations, branch power decreases the rate around 1.5 bits/sec/Hz and the loss due to the correlations is around 0.5 bits/sec/Hz.

It has seen the best local optimum for the algorithm, and how the features of real channel affect the performance of the algorithm.

ANNEX 1 – PROCESSING CHANNELS FOR MATLAB

A1.1 Normalization to a reference handset

```
% Each file contains the coefficients between the two base station(two
antennas) and the handset(one antenna)

% Tx=1:2 belongs to the base station 1 placed in Aalborg destilery
% Tx=3:4 belongs to the base station 2 placed in Aalborg Hospital
Tx = 1:4;
Rx = 1; % Antenna in handset
del = 1:1500; % Delay which the data have been received
Pos = 1:1200; % Different positions which data have been taken

DeltaTau = 2.5e-9; % delay increment
Fs = 60; % IR sampling frequency

% Load Files
% File containing coefficients between base station(all antennas) and
handset1(antenna Rx1)
IR1=load('MBVII-1005/IRDat_Rx17_Data_2.mat');
% File containing coefficients between base station(all antennas) and
handset2(antenna Rx1)
IR3=load('MBVII-1005/IRDat_Rx29_Data_2.mat');

% Extract desired information inside the files
IRtemp1 = IR1.IRDat.IR;
IRtemp2 = IR2.IRDat.IR;
IRtemp3 = IR3.IRDat.IR;

%----------------------------------------------------------------------
%Power Delay Profile
%----------------------------------------------------------------------

% First Antenna at Base station 1 - Antenna at user 1 is chosen
% Power Delay Profile at different handset positions
PDP1=abs(squeeze (IRtemp1(:,1,1,Pos))).^2;

PDP_multi1 = (PDP1(:,300));
PDP_multi2 = (PDP1(:,600));
PDP_multi3 = (PDP1(:,900));
PDP_multi4 = (PDP1(:,1200));

figure

subplot( 2,2,1);
plot(del,10*log10( PDP_multi1 ),'r' )
xlabel( 'Delay' )
ylabel( 'Power [dB]' )
title( 'Power-delay profile at position 300' )

subplot( 2,2,2);
```

```
plot(del,10*log10( PDP_multi2 ),'b' )
xlabel( 'Delay' )
ylabel( 'Power [dB]' )
title( 'Power-delay profile at position 600' )

subplot( 2,2,3);
plot(del,10*log10( PDP_multi3 ),'g' )
xlabel( 'Delay' )
ylabel( 'Power [dB]' )
title( 'Power-delay profile at position 900' )

subplot( 2,2,4);
plot(del,10*log10( PDP_multi4 ),'m' )
xlabel( 'Delay' )
ylabel( 'Power [dB]' )
title( 'Power-delay profile at position 1200' )

%---------------------------------------------------------------------
%---------------------------------------------------------------------

%---------------------------------------------------------------------
%%  Transfer functions
%---------------------------------------------------------------------
% Frequency selection and fast Fourier transformation to get the
coefficients for each handset

                    IR1=squeeze(IRtemp1( :,1:4,1,Pos ));
                    IR3=squeeze(IRtemp3( :,1:4,1,Pos ));

                    TFcmpx1 = fft( IR1, [],1 )./sqrt(1500);
                    TFcmpx3 = fft( IR3, [],1 )./sqrt(1500);

%---------------------------------------------------------------------
%---------------------------------------------------------------------

%---------------------------------------------------------------------
%Extract Coefficients channels
%---------------------------------------------------------------------
% Put coefficients according link where they belong

h1111=(squeeze(TFcmpx1(10,1,:)));
h1112=(squeeze(TFcmpx1(10,2,:)));
h1211=(squeeze(TFcmpx1(10,3,:)));
h1212=(squeeze(TFcmpx1(10,4,:)));
%----------------------------------
h2211=(squeeze(TFcmpx3(10,1,:)));
h2212=(squeeze(TFcmpx3(10,2,:)));
h2111=(squeeze(TFcmpx3(10,3,:)));
h2112=(squeeze(TFcmpx3(10,4,:)));

%---------------------------------------------------------------------
%---------------------------------------------------------------------


%---------------------------------------------------------------------
% Channel Gain --> before normalization
%---------------------------------------------------------------------
% Compute the amplitude of each coefficient and show in dB
```

```
amph1111=(abs(h1111).^2);
amph1112=(abs(h1112).^2);
amph1211=(abs(h1211).^2);
amph1212=(abs(h1212).^2);

amph2111=(abs(h2111).^2);
amph2112=(abs(h2112).^2);
amph2211=(abs(h2211).^2);
amph2212=(abs(h2212).^2);

figure
subplot( 2,1,1);
x=1:1:1200;
plot(x,10*log10(amph1111),'r',x,10*log10(amph1112),'b')
hold on
title('H 11 : BS 1 - User 1')
xlabel( 'Position' )
ylabel( 'Channel gain[dB]' )
legend('h11','h12', -1);

subplot( 2,1,2);
plot(x,10*log10(amph1211),'r',x,10*log10(amph1212),'b')
title('H 12 : BS 2 - User 1')
xlabel( 'Position' )
ylabel( 'Channel gain[dB]' )
legend('h11','h12', -1);

figure
subplot( 2,1,1);

plot(x,10*log10(amph2111),'r',x,10*log10(amph2112),'b')
title('H 21 : BS 1 - User 2')
xlabel( 'Position' )
ylabel( 'Channel gain[dB]' )
legend('h11','h12', -1);

subplot( 2,1,2);
plot(x,10*log10(amph2211),'r',x,10*log10(amph2212),'b')
title('H 22 : BS 2 - User 2')
xlabel( 'Position' )
ylabel( 'Channel gain[dB]' )
legend('h11','h12', -1);

%-------------------------------------------------------------------
%-------------------------------------------------------------------

%-------------------------------------------------------------------
%Channel normalization --> Handset Reference Normalization
%-------------------------------------------------------------------
% Put coefficients according links between Base station - Handset
A1=[h1111 h1112];
A2=[h1211 h1212];
A3=[h2111 h2112];
A4=[h2211 h2212];

% Compute Mean power for reference handset
links_B1_H6=[h1111 h1112];
```

```
links_B1_H6_aux=abs(links_B1_H6).^2;
D1=mean(mean(links_B1_H6_aux));

%--------------------------------------------------------------------
%Normalization each link with reference power
A1_norm=A1./sqrt(D1);
A2_norm=A2./sqrt(D1);
A3_norm=A3./sqrt(D1);
A4_norm=A4./sqrt(D1);
%--------------------------------------------------------------------

%--------------------------------------------------------------------
%Extract normalized coefficients
h1111norm=A1_norm(:,1);
h1112norm=A1_norm(:,2);
h1211norm=A2_norm(:,1);
h1212norm=A2_norm(:,2);

h2111norm=A3_norm(:,1);
h2112norm=A3_norm(:,2);
h2211norm=A4_norm(:,1);
h2212norm=A4_norm(:,2);

%--------------------------------------------------------------------
%--------------------------------------------------------------------



%--------------------------------------------------------------------
%Plots CDF and Probplot per each Link --> H11, H12, H21, H22
%--------------------------------------------------------------------

H11=[A1_norm(:,1)' A1_norm(:,2)'];
H12=[A2_norm(:,1)' A2_norm(:,2)'];
H21=[A3_norm(:,1)' A3_norm(:,2)'];
H22=[A4_norm(:,1)' A4_norm(:,2)'];

H11abs=abs(H11);
H12abs=abs(H12);
H21abs=abs(H21);
H22abs=abs(H22);

figure

subplot(2,2,1)
probplot('rayleigh',H11abs)
title('H11')
subplot(2,2,2)
probplot('rayleigh',H12abs)
title('H12')
subplot(2,2,3)
probplot('rayleigh',H21abs)
title('H21')
subplot(2,2,4)
probplot('rayleigh',H22abs)
title('H22')

figure
```

```
subplot( 2,2,1);
cdfplot(abs(H11abs));
title('CDF Channel H11');

subplot( 2,2,2);
cdfplot(abs(H12abs));
title('CDF Channel H12');

subplot( 2,2,3);
cdfplot(abs(H21abs));
title('CDF Channel H21');

subplot( 2,2,4);
cdfplot(abs(H22abs));
title('CDF Channel H22');

%----------------------------------------------------------------------
%----------------------------------------------------------------------



%----------------------------------------------------------------------
% Channel Gain after normalization
%----------------------------------------------------------------------
% Compute the amplitude of each normalized coefficient and show it in
dB
amph1111norm=(abs(h1111norm).^2);
amph1112norm=(abs(h1112norm).^2);
amph1211norm=(abs(h1211norm).^2);
amph1212norm=(abs(h1212norm).^2);

amph2111norm=(abs(h2111norm).^2);
amph2112norm=(abs(h2112norm).^2);
amph2211norm=(abs(h2211norm).^2);
amph2212norm=(abs(h2212norm).^2);

figure

subplot( 2,1,1);
x=1:1:1200;
plot(x,10*log10(amph1111norm),'r',x,10*log10(amph1112norm),'b')
hold on
title('H 11 : BS 1 - User 1')
xlabel( 'Position' )
ylabel( 'Channel gain[dB]' )
legend('h11','h12', -1);

subplot( 2,1,2);
plot(x,10*log10(amph1211norm),'r',x,10*log10(amph1212norm),'b')
title('H 12 : BS 2 - User 1')
xlabel( 'Position' )
ylabel( 'Channel gain[dB]' )
legend('h11','h12', -1);

figure

subplot( 2,1,1);
```

```
plot(x,10*log10(amph2111norm),'r',x,10*log10(amph2112norm),'b')
title('H 21 : BS 1 - User 2')
xlabel( 'Position' )
ylabel( 'Channel gain[dB]' )
legend('h11','h12', -1);

subplot( 2,1,2);
plot(x,10*log10(amph2211norm),'r',x,10*log10(amph2212norm),'b')
title('H 22 : BS 2 - User 2')
xlabel( 'Position' )
ylabel( 'Channel gain[dB]' )
legend('h11','h12', -1);

%---------------------------------------------------------------------
%---------------------------------------------------------------------

%---------------------------------------------------------------------
% Plot histogram PDF Rayleigh per each Link --> H11, H12, H21, H22
%---------------------------------------------------------------------

figure
subplot(2,2,1);
hist(H11abs,100);
title('Histogram normalized samples H11');

subplot(2,2,2);
hist(H12abs,100);
title('Histogram normalized samples H12');

subplot(2,2,3);
hist(H21abs,100);
title('Histogram normalized samples H21');

subplot(2,2,4);
hist(H22abs,100);
title('Histogram normalized samples H22');

%---------------------------------------------------------------------
%---------------------------------------------------------------------

%---------------------------------------------------------------------
%  Plot cross correlation between each coefficient for each Base
station
%---------------------------------------------------------------------
figure

subplot( 2,2,1);
crosscorr(abs(h1111norm),abs(h1112norm),min([length(abs(h1111norm)),le
ngth(abs(h1112norm))])-1);
title('Cross-Correlation coefficients BS1 - User 1');
ylim([0 1]);


subplot( 2,2,2);
crosscorr(abs(h1211norm),abs(h1212norm),min([length(abs(h1211norm)),le
ngth(abs(h1212norm))])-1);
title('Cross-Correlation coefficients BS2 - User 1');
```

```
ylim([0 1]);


subplot( 2,2,3);
crosscorr(abs(h2111norm),abs(h2112norm),min([length(abs(h2111norm)),le
ngth(abs(h2112norm))])-1);
title('Cross-Correlation coefficients BS1 - User 2');
ylim([0 1]);


subplot( 2,2,4);
crosscorr(abs(h2211norm),abs(h2212norm),min([length(abs(h2211norm)),le
ngth(abs(h2212norm))])-1);
title('Cross-Correlation coefficients BS2 - User 2');
ylim([0 1]);

%---------------------------------------------------------------------
%---------------------------------------------------------------------

%---------------------------------------------------------------------
% Plot autocorrelation each coefficient
%---------------------------------------------------------------------
%Coefficients for handset 1
figure

subplot( 2,2,1);
autocorr(abs(h1111norm),length(abs(h1111norm))-1);
title('Auto-Correlation BS1-User1 Tx1-Rx1 ');
ylim([0 1]);

subplot( 2,2,2);
autocorr(abs(h1112norm),length(abs(h1112norm))-1);
title('Auto-Correlation BS1-User1 Tx2-Rx1');
ylim([0 1]);

subplot( 2,2,3);
autocorr(abs(h1211norm),length(abs(h1211norm))-1);
title('Auto-Correlation BS2-User1 Tx1-Rx1');
ylim([0 1]);

subplot( 2,2,4);
autocorr(abs(h1212norm),length(abs(h1212norm))-1);
title('Auto-Correlation BS2-User1 Tx2-Rx1');
ylim([0 1]);

%Coefficients for handset 2
figure

subplot( 2,2,1);
autocorr(abs(h2211norm),length(abs(h2211norm))-1);
title('Auto-Correlation BS1-User2 Tx1-Rx1');
ylim([0 1]);

subplot( 2,2,2);
autocorr(abs(h2212norm),length(abs(h2212norm))-1);
title('Auto-Correlation BS1-User2 Tx2-Rx1');
ylim([0 1]);
```

```
subplot( 2,2,3);
autocorr(abs(h2111norm),length(abs(h2111norm))-1);
title('Auto-Correlation BS2-User2 Tx1-Rx1');
ylim([0 1]);

subplot( 2,2,4);
autocorr(abs(h2112norm),length(abs(h2112norm))-1);
title('Auto-Correlation BS2-User2 Tx2-Rx1');
ylim([0 1]);

%-----------------------------------------------------------------------
%-----------------------------------------------------------------------

%-----------------------------------------------------------------------
% Put coefficient into a big matrix Hall. %Hall(:,:,1) --> 1 channel
%-----------------------------------------------------------------------

H_aux_11=zeros(1,2);
H_aux_12=zeros(1,2);
H_aux_21=zeros(1,2);
H_aux_22=zeros(1,2);
Hall=cell(2,2,1200);


for p=1:1200

    H_aux_11(1,1)=h1111norm(p);
    H_aux_11(1,2)=h1112norm(p);


    H_aux_12(1,1)=h1211norm(p);
    H_aux_12(1,2)=h1212norm(p);


    H_aux_21(1,1)=h2111norm(p);
    H_aux_21(1,2)=h2112norm(p);


    H_aux_22(1,1)=h2211norm(p);
    H_aux_22(1,2)=h2212norm(p);

    Hall{1,1,p}=H_aux_11;
    Hall{1,2,p}=H_aux_12;
    Hall{2,1,p}=H_aux_21;
    Hall{2,2,p}=H_aux_22;

end

%Save Data
save('Real_Channels.mat','Hall');

%-----------------------------------------------------------------------
%-----------------------------------------------------------------------
```

A1.2 Normalization to each link

The code of this normalization is the same as the code shown in A.1.1, but changing the normalization part. It is only shown the part which has been changed.

```
%---------------------------------------------------------------------
%Channel normalization --> Normalization each coefficient
%---------------------------------------------------------------------

    h1111norm=h1111./sqrt(mean(abs(h1111).^2));
    h1112norm=h1112./sqrt(mean(abs(h1112).^2));

    h1211norm=h1211./sqrt(mean(abs(h1211).^2));
    h1212norm=h1212./sqrt(mean(abs(h1212).^2));


    h2111norm=h2111./sqrt(mean(abs(h2111).^2));
    h2112norm=h2112./sqrt(mean(abs(h2112).^2));


    h2211norm=h2211./sqrt(mean(abs(h2211).^2));
    h2212norm=h2212./sqrt(mean(abs(h2212).^2));


%---------------------------------------------------------------------
%---------------------------------------------------------------------
```

A1.3 Removing cross correlation

The code of this normalization is the same as the code shown in A.1.1, for the purpose to keep the branch power ratio between each link. It has been explained in chapter 5 that cross correlation only affects at base station 2. To remove this effect, it has been moved one of the coefficients to avoid the cross correlation in lag=0.

Extract coefficients have been only changed respect the A.1.1 code.

```
%----------------------------------------------------------------------
%Extract Coefficients channels
%----------------------------------------------------------------------

h1111=(squeeze(TFcmpx1(10,1,:)));
h1112=(squeeze(TFcmpx1(10,2,:)));
h1211=(squeeze(TFcmpx1(10,3,:)));
h1212=(squeeze(TFcmpx1(10,4,:)));
%Move coefficients 700 positions in a circular way
h1212=circshift(h1212,700);

h2111=(squeeze(TFcmpx3(10,1,:)));
h2112=(squeeze(TFcmpx3(10,2,:)));
h2211=(squeeze(TFcmpx3(10,3,:)));
h2212=(squeeze(TFcmpx3(10,4,:)));
%Move coefficients 700 positions in a circular way
h2212=circshift(h2212,700);

%----------------------------------------------------------------------
%----------------------------------------------------------------------
```

ANNEX 2 – ALGORITHMS FOR MATLAB

Zero-Forcing algorithm and Matched-Filter algorithm have been developed for the environment which we use; it can be seen at figure 2.6.

A2.1 WMMSE-WSR

The first code shows WMMSE – WSR with $B_k^{init} = \boldsymbol{I}_{N*M}$. The others initializations are the same but changing the first part of the code called `% Initialization of Bk`.

A2.1.1 WMMSE-WSR-transmit filter initialization: identity

```
function [WSRalg] =
WSR_EYEinit_average_channels_ok(Hki,Iterations,Wweights,K)
% Number of Tx and Rx antennas from channel dimensions
[N,M] = size(Hki{1,1});

% Initialization of Bk, tx filter
for k =1:K
    Bk(:,:,k) = eye(M,N);
end

% Initialization Power
for k=1:K
    P(k)=1;
end

WSRalg = zeros(1,Iterations);

% Start algorithm
for n=1:Iterations,

    %Compute Effective Noise Covariance Matrix
    for k=1:K
        for i=1:K
            if i~=k
               Rvv(:,:,k)= Hki{k,i}*Bk(:,:,i)*Bk(:,:,i)'*Hki{k,i}';
            end
        end
        Rvv(:,:,k)= Rvv(:,:,k) + eye(N);
    end

    %Update Receive Filter. A_MMSE.
    for k=1:K

Ak(:,:,k)=Bk(:,:,k)'*Hki{k,k}'*pinv(Hki{k,k}*Bk(:,:,k)*Bk(:,:,k)'*Hki{
k,k}' + Rvv(:,:,k));
    end
```

```
    %Update MSE, Weights belongs to WMMSE
    for k=1:K
        Einv(:,:,k)= eye(N) +
Bk(:,:,k)'*Hki{k,k}'*pinv(Rvv(:,:,k))*Hki{k,k}*Bk(:,:,k);
    end

    for k=1:K
        Wk(:,:,k)=Wweights(k)*Einv(:,:,k);
    end

    %Update lambdas
    %Compute Tk
    for k=1:K
        Tk(:,:,k)=zeros(M);
        for i=1:K
            Tk(:,:,k) = Tk(:,:,k) +
Hki{i,k}'*Ak(:,:,i)'*Wk(:,:,i)*Ak(:,:,i)*Hki{i,k};
        end
    end

    %Compute Singular Value Descomposition for Tk
    for k=1:K
        [U,S] = svd(Tk(:,:,k));
        Uk(:,:,k)=U;
        Sk(:,:,k)=S;
    end

    %Compute Gk
    for k=1:K
        Gk(:,:,k) =
Uk(:,:,k)'*Hki{k,k}'*Ak(:,:,k)'*Wk(:,:,k)*Wk(:,:,k)'*
Ak(:,:,k)*Hki{k,k}*Uk(:,:,k);
    end

    % Coeficients for calculate lambda and update lambda
    for k=1:K
        for m=1:M
            gm(m)=real(Gk(m,m,k));
            sm(m)=real(Sk(m,m,k));
        end
        options = optimset('Display','off');  % Turn off display
        lambda_solve(k)=fsolve(@(lambda)
lambdaeq(lambda,k,gm,sm,M,P),0,options);

        if real(lambda_solve(k)) == lambda_solve(k) &&
lambda_solve(k)>0
           lambda_solve(k)=real(lambda_solve(k));
            else
                lambda_solve(k)=0;
            end
    end

    %Update Transmit Filters
    for k=1:K
        Bk(:,:,k) = pinv(Tk(:,:,k) +
lambda_solve(k)*eye(M))*Hki{k,k}'*Ak(:,:,k)'*Wk(:,:,k);
        alfa = trace(Bk(:,:,k)*Bk(:,:,k)');
        if alfa > 1.01
```

```
            Bk(:,:,k)=sqrt(1/alfa)*Bk(:,:,k); %Re-scale Tx. Filter
        end
    end


    %% Compute Effective Noise Covariance Matrix
    for k=1:K
        for i=1:K
            if i~=k
               Rvv(:,:,k)= Hki{k,i}*Bk(:,:,i)*Bk(:,:,i)'*Hki{k,i}';
            end
        end
        Rvv(:,:,k)= Rvv(:,:,k) + eye(N);
    end

    %% update MSE
    for k=1:K
      Ek(:,:,k)= pinv(eye(N) +
      Bk(:,:,k)'*Hki{k,k}'*pinv(Rvv(:,:,k))*Hki{k,k}*Bk(:,:,k));
    end

    %Update Rate_k
    for k=1:K
        Rate(k)=real(log2(det(pinv(Ek(:,:,k)))));
    end

    %Sum-Rate each iteration to plot convergence.
     for k=1:K
         WSRalg(n)= WSRalg(n) + Wweights(k)*Rate(k);
     end



end

end

%Function to do lambda equations
function y = lambdaeq(lambda,k,gm,sm,M,P)

ztot=0;

for m=1:M
    z(m)=gm(m)/(lambda+sm(m))^2;
    ztot = ztot + z(m);
end



y = ztot - P(k);

end
```

A2.1.2 WMMSE-WSR-transmit filter initialization: matched-filter

Initialization with matched-filter:

```
% Initialization of Bk
 for k=1:K,
     Bk(:,:,k) = Hki{k,k}'; %transmit matched filter
     x(k)=trace(Bk(:,:,k)*Bk(:,:,k)');
     Bk(:,:,k)=sqrt(P(k)/x(k))*Bk(:,:,k);
end
```

A2.1.3 WMMSE-WSR-transmit filter initialization: zero-forcing

Initialization with zero-forcing:

```
%Initialization of Bk
for k =1:K
    Bk(:,:,k) = zeros(M,N);
end

%Init Bk with ZF values

for k=1:K
    for i=1:K
        if i~=k
           Ortho(:,:,i) = eye(M) - Hki{k,i}'*pinv( Hki{k,i}*Hki{k,i}'
)*Hki{k,i};
        end
    end
end

for k=1:K
    Bk(:,:,k)= Ortho(:,:,k)*Hki{k,k}';
    x(k)=trace(Bk(:,:,k)*Bk(:,:,k)');
    Bk(:,:,k)=sqrt(P(k)/x(k))*Bk(:,:,k);
end
```

A2.1.4 WMMSE-WSR-transmit filter initialization: 10 random initializations

In this case algorithm has been run 10 times with 10 random initializations which are a enter parameter.

```
for random=1:10
    B_init = randn(Ntx,Nrx,K) + i*randn(Ntx,Nrx,K);
    for k=1:K
        x(k)=trace(B_init(:,:,k)*B_init(:,:,k)');
       B_init(:,:,k)=sqrt(1/x(k))*B_init(:,:,k);
   end
end
```

A2.2 Zero-forcing

```
Function[Ratesum]=ZEROF_WSR_average_channels_ok(Hki,Wweights,K)
[N,M] = size(Hki{1,1});
Ratesum=0;
 %Initialization of Bk
for k =1:K
    Bk(:,:,k) = zeros(M,N);
end
 %Initialization Power
for k=1:K
    P(k)=1;
end

%Compute Zero Forcing transmitted filters
for k=1:K
    for i=1:K
        if i~=k
  Ortho(:,:,i) = eye(M) - Hki{k,i}'*pinv(Hki{k,i}*Hki{k,i}')*Hki{k,i};
        end
    end
end

for k=1:K
    Bk(:,:,k)= Ortho(:,:,k)*Hki{k,k}';
    x(k)=trace(Bk(:,:,k)*Bk(:,:,k)');
    Bk(:,:,k)=sqrt(P(k)/x(k))*Bk(:,:,k);
end

%Compute Effective Noise Covariance Matrix
for k=1:K
    for i=1:K
        if i~=k
           Rvv(:,:,k)= Hki{k,i}*Bk(:,:,i)*Bk(:,:,i)'*Hki{k,i}';
        end
    end
    Rvv(:,:,k)= Rvv(:,:,k) + eye(N);
end

%Compute I + SINR
for k=1:K
    Ek_inv(:,:,k)= eye(N) +
Bk(:,:,k)'*Hki{k,k}'*pinv(Rvv(:,:,k))*Hki{k,k}*Bk(:,:,k);
end

%Update Rate_k

for k=1:K
    Rate(k)=real(log2(det(Ek_inv(:,:,k))));
end

%Sum Rates
for k=1:K
    Ratesum= Ratesum + Rate(k);
end
end
```

A2.3 Matched-filter

```
function [Ratesum] =
MATCHED_WSR_average_channels_ok(Hki,Iterations,Wweights,K)
[N,M] = size(Hki{1,1});
Ratesum=0;

%Initialization of Bk
for k =1:K
Bk(:,:,k) = zeros(M,N);
end

%Power Constraint
for k =1:K
P(k)=1;
end



%Transmit Matched Filter
for k=1:K,
    Bk(:,:,k) = Hki{k,k}';
    x(k)=trace(Bk(:,:,k)*Bk(:,:,k)');
    Bk(:,:,k)=sqrt(P(k)/x(k))*Bk(:,:,k);
end;

%Compute Effective Noise Covariance Matrix
for k=1:K
    for i=1:K
        if i~=k
           Rvv(:,:,k)= Hki{k,i}*Bk(:,:,i)*Bk(:,:,i)'*Hki{k,i}';
        end
    end
    Rvv(:,:,k)= Rvv(:,:,k) + eye(N);
end

%Compute I + SINR
for k=1:K
    Ek_inv(:,:,k)= eye(N) +
(Hki{k,k}*Bk(:,:,k)*Bk(:,:,k)'*Hki{k,k}')/Rvv(:,:,k);
end

%Update Rate_k
for k=1:K
    Rate(k)=real(log2(det(Ek_inv(:,:,k))));
end

%Sum Rates
for k=1:K
    Ratesum= Ratesum + Rate(k);
end

end
```

REFERENCES

[1] G.J. FOSCHINI and M.J. GANS," On Limits of Wireless Communications in a Fading Environment when Using Multiple Antennas", Wireless Personal Communications pp.311–335, 1998.

[2] E. Telatar, "Capacity of Multi-antenna Gaussian Channels," European Transactions on Telecommunications, vol. 10, no. 6, pp. 585-595, 2000.

[3] M. K. Karakayali, G. J. Foschini, and R.A. Valenzuela, "Network coordination for spectrally efficient communications in cellular systems," IEEE Wireless Communications, vol. 13, no. 4, pp. 56-61, 2006.

[4] M. Chiani, M. Win and H.Shin, "MIMO Networks: The Effects of Interference ," IEEE Trans. & Info Theory. Proc., pp. 336–349, 2010.

[5] S.S. Christensen, R. Agarwal, E. Carvalho, and J. Cioffi, "Weighted sumrate maximization using weighted MMSE for MIMO-BC beamforming design," IEEE Trans. on Wireless Communications, vol. 7, no. 12, pp. 4792–4799, 2008.

[6] Francesco Negro, Shakti Prasad Shenoy, Irfan Ghauri, Dirk T.M. Slock, "On the MIMO interference channel" Information Theory and Applications Workshop (ITA),pp. 1-9,2010.

[7] Yanakiev, Boyan; Nielsen, Jesper O.; Pedersen, Gert F. " On small antenna measurements in a realistic MIMO scenario", Antennas and Propagation (EuCAP), pp 1 -5, 2010.

[8] Jarmyr, S.; Ottersten, B.; Jorswieck, E.; "Statistical Precoding and Detection Ordering in MIMO Multiple-Access Channels with Decision Feedback Equalization", IEEE International Conference on Communications (ICC), June 2011.

[9] Sigdel, S.; Krzymien, W.A.; "Efficient User Selection and Ordering Algorithms for Successive Zero-Forcing Precoding for Multiuser MIMO Downlink", Vehicular Technology Conference, pp 1 – 6, 2009.

[10] Kaare Brandt Pedersen, Michael Syskind Pedersen; "The Matrix Cookbook", Version: November 14, 2008

[11] Hui Shen; Bin Li; Meixia Tao; Xiaodong Wang; "MSE-Based Transceiver Designs for the MIMO Interference Channel", IEEE Transactions on Wireless Communications pp 3480 – 3489, 2010.

[12] Verde, F.; Darsena, D.; Scaglione, A.; "Cooperative Randomized MIMO-OFDM Downlink for Multicell Networks: Design and Analysis", IEEE Transactions on Signal Processing, pp 384 – 402, 2010

[13] Dabora, R.; Servetto, S.D.; "Broadcast Channels with Cooperating Decoders", IEEE Tansactions on Information Theory, pp 5438 – 5454, 2006.

[14] Yanakiev, Boyan; Christensen, Morten; Pedersen, Gert F. "Using Optical Links for Radio Propagation Measurements – Part 2", EURO-COST, pp 1 – 6, 2010.

[15] Nielsen, J. O.; Yanakiev, B.; Bonev, I. B.; Christensen, M.; Pedersen, G. F.; "User Influence on MIMO Channel Capacity for Handsets in Data Mode Operation" IEEE Transactions on Antennas and Propagation, pp 1 – 11, 2011.

[16] Matlab Documentation Functions [online] "www.matworks.com/help"