TECHNICAL UNIVERSITY OF CRETE
ELECTRICAL AND COMPUTER ENGINEERING DEPARTMENT
TELECOMMUNICATIONS DIVISION

Asynchronous Multi-User Detection using Inference Algorithms

by

Tsetis Ioannis

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DIPLOMA OF

ELECTRICAL AND COMPUTER ENGINEERING

November 2017

THESIS COMMITTEE

Associate Professor Aggelos Bletsas, *Thesis Supervisor*
Professor Michael Paterakis
Associate Professor George N. Karystinos

Abstract

This work highlights state-of-the-art probabilistic graphical models and inference algorithms that exploit rather than avoid the symbol-level asynchrony in multi-user wireless communications. More specifically, the recently proposed inference algorithm SigSag is studied, where multiple users transmit their data packets at the same time and frequency channels, with however random delays; assuming channel state information (CSI) at the receiver, linear equations are formed, which produce a probabilistic graphical model (PGM), amenable to inference algorithms. This work implements the sum-product belief propagation algorithm on the crafted PGM and a) derives the message passing equations, b) studies initialization and c) complements with CSI estimation, using linear minimum mean squared error (LMMSE) estimator. Performance was tested for 2 or 3 users. It was found that performance was sensitive to initialization, as expected, due to the inherently loopy nature of the crafted PGM for small packet lengths. On the contrary, bit error rate (BER) decreases with increasing packet length, at the expense of convergence time. Moreover, reduced convergence time results to higher BER.

Acknowledgements

First of all I would like to thank my supervisor Prof. Aggelos Bletsas for his guidance and support throughout this work and beyond that.

I would also like to express gratitude to my committee, that consists of Prof. Michael Paterakis and Prof. George N.Karystinos. Both of these professors were excellent in their respective classes.

Furthermore special thanks are given to PhD holder, Panos Alevizos for all the assistance he gave me in this thesis and for being a great person in general.

Additionally i would like to thank my close friends and all the laboratory members who made my time studying in Chania easier and smoother.

Lastly and more importantly i would like to thank my family and girlfriend who supported me in every way possible throughout these past years.

Table of Contents

List of Figures

Chapter 1

Introduction and System Model

1.1 State of the art multi-user communication

There has been a tremendous amount of theoretical work in multi-user detection and interference cancellation, where most implementations rely on carrier sense multiple access (CSMA) to limit collisions while code-division multiple access (CDMA) receivers decode each user by treating interference as noise.

A different innovative approach by [1] proved that probabilistic graphical models and inference algorithms can be a great tool for performing multi-user joint decoding and interference cancellation. Another important step towards more practical systems that decode interfering users was ZigZag decoding by [2], that exploited the asynchrony across succesive collisions. After that, based on the same assumptions as the original ZigZag framework SigSag, a soft-decoding version was developed by [3].

In this work, it is assumed that there are N users, each wanting to transmit a packet of B bits, trying to communicate with an access point (AP) where each user relies on carrier sensing to detect if other users are transmitting. If this method fails, there is interference at the AP, modelled by a simple linear superposition of the symbols plus noise.

The focus of this work is on the worst-case scenario where carrier sensing constantly fails and packet collisions are always formed in the AP. In that case, the model that is used assumes that each of the N users transmits its packet N times and the AP receives linear equations involving the sum of the collided symbols plus noise.

The problem that is being solved in this work, is a maximum likelihood detection problem that consists of finding the most probable user symbols given the noise statistics. In the high-SNR case(when the noise is negligible) this simply reduces to solving linear equations, while for the noisy case it becomes a statistical inference problem which is well known to be computationally intractable.

1.2 Lack of Symbol-level synchronization

As mentioned by [2] there are a few practical issues that should be mentioned because they complicate the process of estimating the transmitted symbols from the received symbols. Two core problems that should be mentioned, are sampling offset and inter-symbol interference.

a) Sampling Offset: The transmitted signal is a sequence of complex valued samples separated by a period T. However when they are transmitted on the wireless medium, these discrete values have to be interpolated into a continuous signal. The continuous signal is equal to the original discrete samples, only if they are sampled at the exact same positions where the discrete values were. Due to general lack of synchronization a receiver will not be able to sample the received signal exactly at the right positions and there will always be a sampling offset μ. This offset originates from the drift in the receiver and transmitter clock. Hence, decoders have algorithms to estimate and keep track of it over the duration of a packet.

b) Inter-Symbol interference:In a fully-realistic scenario there will be inter-symbol interference because the symbol duration will not be the same and the received symbol y[n] will not just be a product of the transmitted symbol x[n] but in practise neighboring symbols may interfere with it.

In the system model presented in this work, for simplicity's sake it was assumed that symbol duration is always the same and the delay of each user is an integer multiple of the slots of the transmission.

1.3 General System Model

In the system model of this work, there are N users trying to communicate to an access point. Each user re-transmits N times and the packets collide forming linear equations at the access point. A block Rayleigh fading channel model is assumed where data sent by user i on the c^{th} transmission are attenuated by coefficients $h_i^{(c)}$ that are assumed known at the receiver. For the sake of completeness in 3 a way to estimate the fading coefficients $h_i^{(c)}$ at the receiver by using a known preamble sequence to each packet is presented. So, for this estimation to be exact, the fading is required to be extremely slow. Under this model, the access point receives the signal:

$$\vec{u_c} = \sum_{i=1}^{N} h_i^{(c)} T_{w(i,c)}(\vec{x_i}) + \vec{v^{(c)}}, \tag{1.1}$$

where $c \in (1, 2, ..., N)$ is the collision round and $\vec{x_i} = [x_{i,1}\ x_{i,2}\ ...\ x_{i,B}]$ is the packet (assuming BPSK, $x_{i,j} = \pm 1$) sent by the i^{th} user with B bits packet length. Also $v^{(c)}$ is the channel noise vector of the c^{th} collision which is assumed to be independent and identically distributed complex Gaussian noise. The access point receives the noisy data on the c^{th} collision noted as $\vec{u_c}$. Finally, $T_{w(i,c)}(\vec{x_i})$ is an operator that takes the B dimensional vector $\vec{x_i}$ and creates a B+W dimensional vector by padding zeros to the beginning and the end, where the number of padded zeros at the beginning is determined by the time delay i chooses randomly before the c^{th} transmission. It is assumed that the time delay is known at the access.

So considering all of the above the whole system can be modelled as $A\vec{x} + v = \vec{u}$ where A is the collision matrix, $\vec{x} = [x_1\ x_2\ ...\ x_N]^T$ and $\vec{u} = [u_1\ u_2\ ...\ u_N]^T$. For the aforementioned system a factor graph is designed as it is explained in chapter 2.1.3 and proved in 6.1.

1.3.1 System Model for N=2 Users

In this part, a thorough example of the system model for 2 users is presented that considers all the things mentioned in 1.3. To start with because there are 2 users, each user re-transmits each packet 2 times and the packets collide forming linear equations at the access point. In this example it is considered that the 2 Users are transmitting packets that have 5 bits each like in the

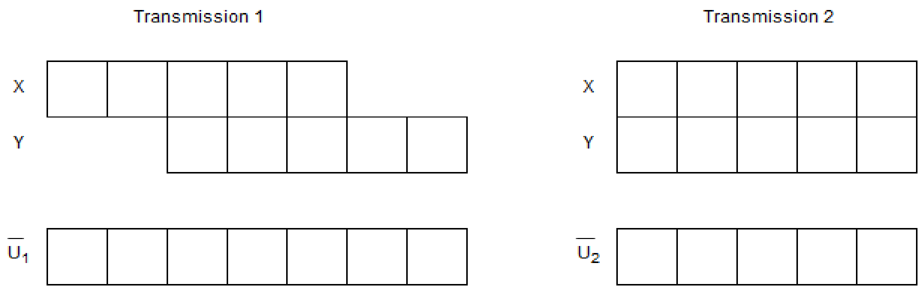

Figure 1.1: Two consecutive collisions of two packets $\vec{x}$ and $\vec{y}$ of B=5 symbols sent by N=2 users. Both packets are transmitted twice and the access point receives $\vec{u_1}$ and $\vec{u_2}$

figure 1.1, so the set of linear equations corresponding to the collision patterns shown in the figure 1.1 are:

$$
\begin{aligned}
u_{11} &= h_1^{(1)}x_1 + v_1^{(1)}\\
u_{12} &= h_1^{(1)}x_2 + v_2^{(1)}\\
u_{13} &= h_1^{(1)}x_3 + h_2^{(1)}y_1 + v_3^{(1)}\\
u_{14} &= h_1^{(1)}x_4 + h_2^{(1)}y_2 + v_4^{(1)}\\
u_{15} &= h_1^{(1)}x_5 + h_2^{(1)}y_3 + v_5^{(1)}\\
u_{16} &= h_2^{(1)}y_4 + v_6^{(1)}\\
u_{17} &= h_2^{(1)}y_5 + v_7^{(1)}\\
u_{21} &= h_1^{(2)}x_1 + h_2^{(2)}y_1 + v_1^{(2)}\\
u_{22} &= h_1^{(2)}x_2 + h_2^{(2)}y_2 + v_2^{(2)}\\
u_{23} &= h_1^{(2)}x_3 + h_2^{(2)}y_3 + v_3^{(2)}\\
u_{24} &= h_1^{(2)}x_4 + h_2^{(2)}y_4 + v_4^{(2)}\\
u_{25} &= h_1^{(2)}x_5 + h_2^{(2)}y_5 + v_5^{(2)}
\end{aligned}
$$

and the corresponding collision matrix A as it is mentioned in 1.3 is the following:

$$\begin{bmatrix} h_1^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & h_1^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & h_1^{(1)} & 0 & 0 & h_2^{(2)} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_1^{(1)} & 0 & 0 & h_2^{(2)} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & h_1^{(1)} & 0 & 0 & h_2^{(2)} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & h_2^{(2)} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & h_2^{(2)} \\ h_1^{(2)} & 0 & 0 & 0 & 0 & h_2^{(2)} & 0 & 0 & 0 & 0 \\ 0 & h_1^{(2)} & 0 & 0 & 0 & 0 & h_2^{(2)} & 0 & 0 & 0 \\ 0 & 0 & h_1^{(2)} & 0 & 0 & 0 & 0 & h_2^{(2)} & 0 & 0 \\ 0 & 0 & 0 & h_1^{(2)} & 0 & 0 & 0 & 0 & h_2^{(2)} & 0 \\ 0 & 0 & 0 & 0 & h_1^{(2)} & 0 & 0 & 0 & 0 & h_2^{(2)} \end{bmatrix} \tag{1.2}$$

In this problem, if there was no noise ($\vec{v^{(c)}} = 0$), the optimal decoder would simply have to solve these linear equations.However, in the presence of noise, optimal decoding would correspond to finding which vectors $\vec{x} \in (\pm 1)^B$, $\vec{y} \in (\pm 1)^B$ have the highest likelihood under the noise statistics, which for general N reduces to a computationally intractable integer least-square problem.

So the approach to solve this problem, was to create a probabilistic graphical model based on the equations and the collision matrix and then use a message-passing algorithm to find the most likely transmitted symbols for user x and y. The probabilistic graphical model that was crafted considering the figure in 1.1 is the following:

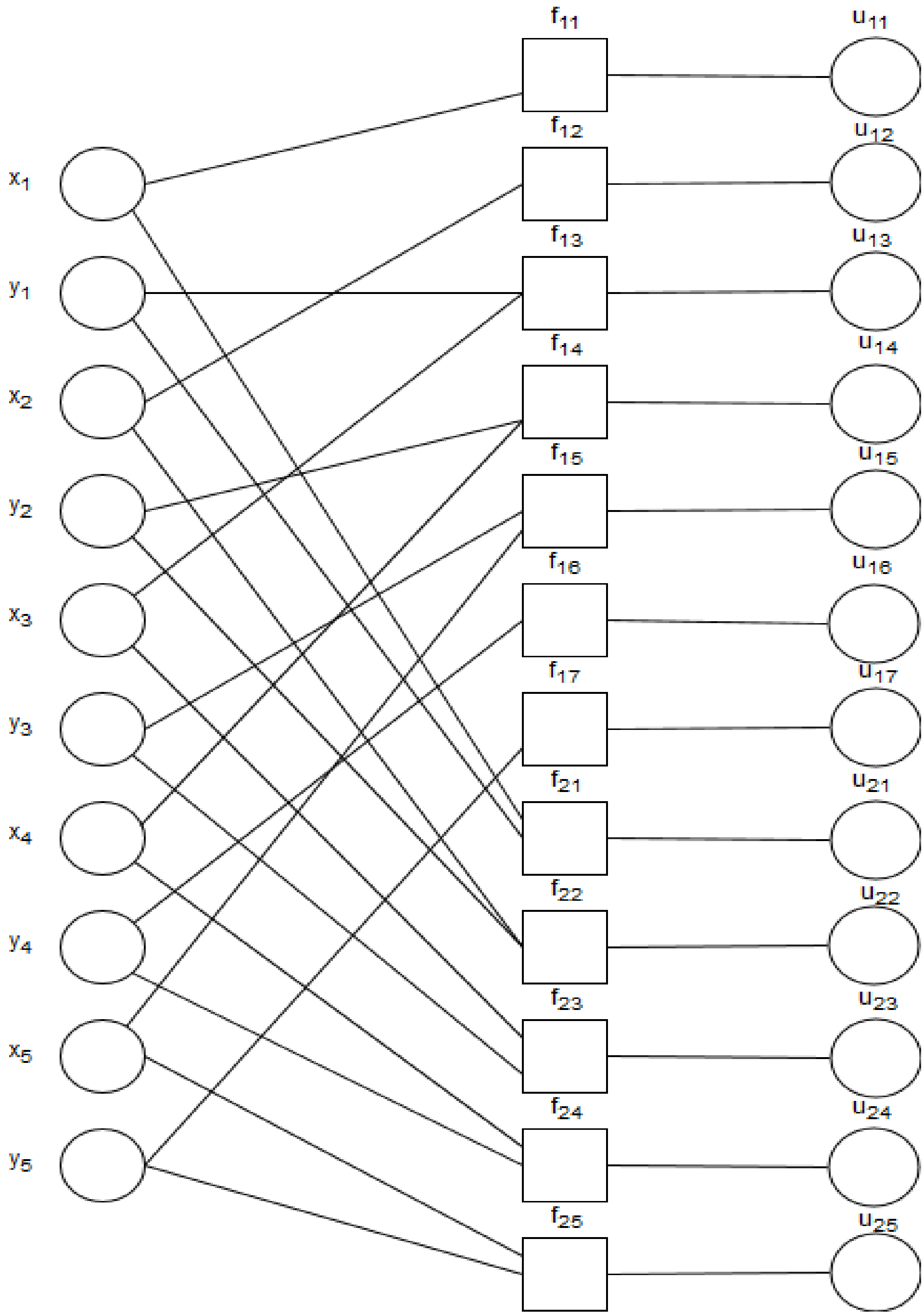

Figure 1.2: Factor graph based on Figure 2.1

After that the factor graph is simplified by applying the factor graph design equations that are mentioned in 2.1.3 and then proved in 6.1.

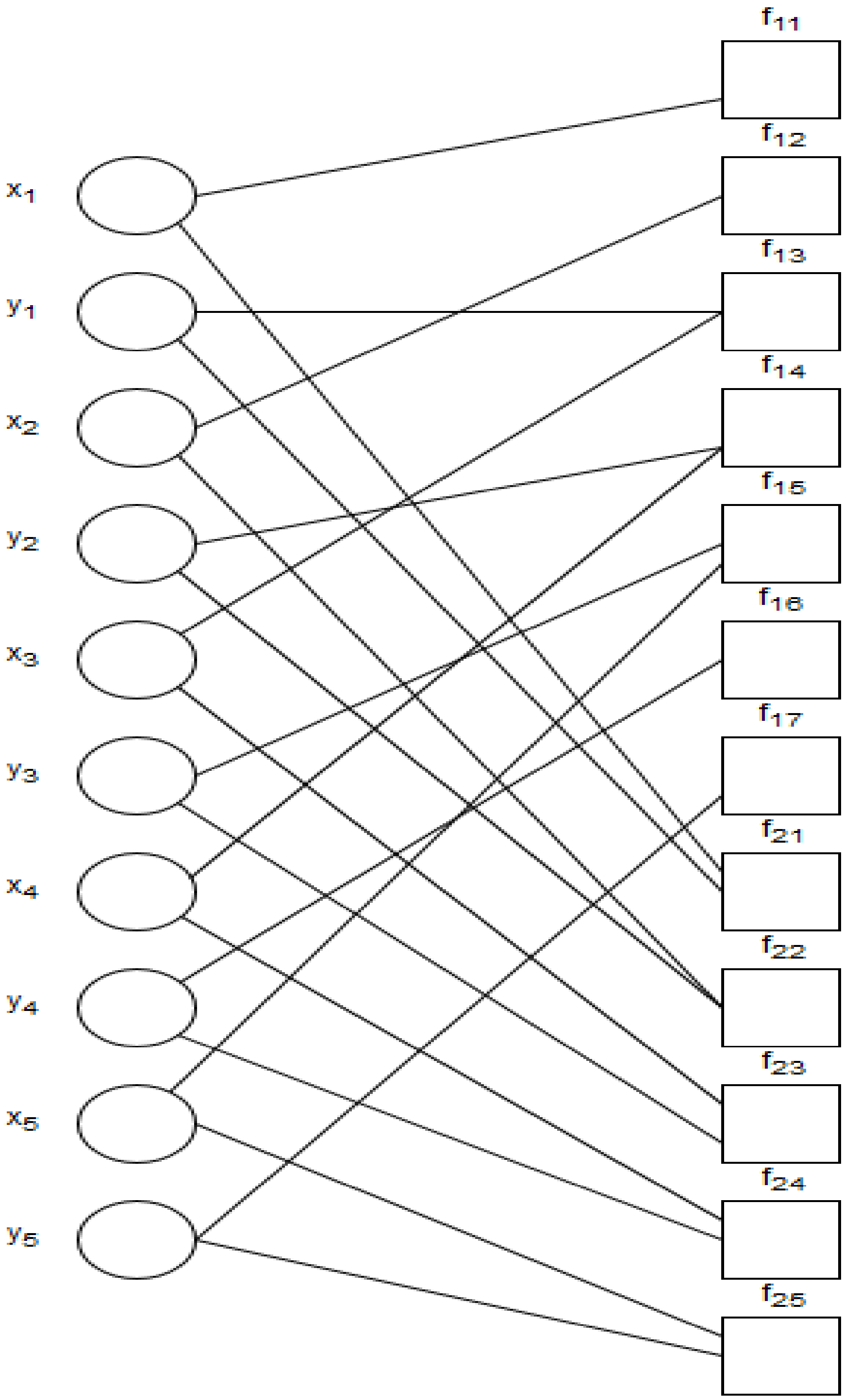

Figure 1.3: Simplified factor graph based on Figure 2.1 after Factor Node Design equations

1.3.2 System Model for N=3 Users

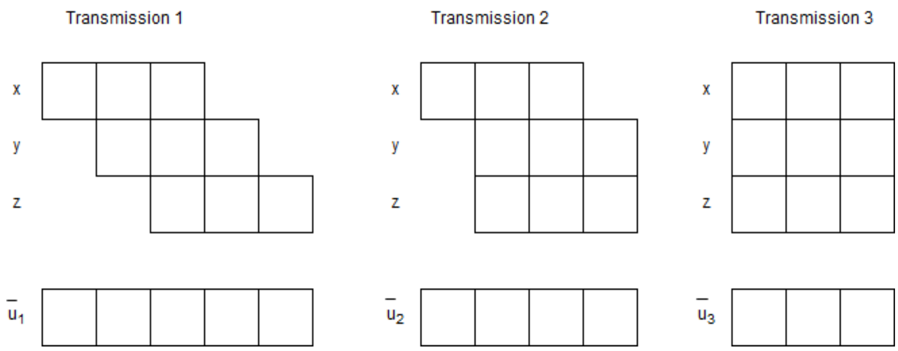

Figure 1.4: Three consecutive collisions of three packets $\vec{x}$, $\vec{y}$ and $\vec{z}$ of B=3 symbols sent by N=3 users. The packets are transmitted thrice and the access point receives $\vec{u_1}$, $\vec{u_2}$ and $\vec{u_3}$

In this part, a thorough example of the system model for 3 users is presented that considers all the things mentioned in 1.3. To start with because there are 3 users , each user re-transmits each packet 3 times and the packets collide forming linear equations at the access point. In this example it is considered that the 3 Users are transmitting packets that have 3 bits each like in the figure 1.4, so the set of linear equations corresponding to the collision patterns shown in the figure 1.4 are:

$$
\begin{aligned}
u_{11} &= h_1^{(1)} x_1 + v_1^{(1)} \\
u_{12} &= h_1^{(1)} x_2 + h_2^{(1)} y_1 + v_2^{(1)} \\
u_{13} &= h_1^{(1)} x_3 + h_2^{(1)} y_2 + h_3^{(1)} z_1 + v_3^{(1)} \\
u_{14} &= h_2^{(1)} y_3 + h_3^{(1)} z_2 + v_4^{(1)} \\
u_{15} &= h_3^{(1)} z_3 + v_5^{(1)} \\
u_{21} &= h_1^{(2)} x_1 + v_1^{(2)} \\
u_{22} &= h_1^{(2)} x_2 + h_2^{(2)} y_1 + h_3^{(2)} z_1 + v_2^{(2)} \\
u_{23} &= h_1^{(2)} x_3 + h_2^{(2)} y_2 + h_3^{(2)} z_2 + v_3^{(2)} \\
u_{24} &= h_2^{(2)} y_3 + h_3^{(2)} z_3 + v_4^{(2)} \\
u_{31} &= h_1^{(3)} x_1 + h_2^{(3)} y_1 + h_3^{(3)} z_1 + v_1^{(3)} \\
u_{31} &= h_1^{(3)} x_2 + h_2^{(3)} y_2 + h_3^{(3)} z_2 + v_2^{(3)} \\
u_{31} &= h_1^{(3)} x_3 + h_2^{(3)} y_3 + h_3^{(3)} z_3 + v_3^{(3)}
\end{aligned}
$$

and the corresponding collision matrix A as it is mentioned in 1.3 is the following:

$$
\begin{bmatrix}
h_1^{(1)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & h_1^{(1)} & 0 & h_2^{(1)} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & h_1^{(1)} & 0 & h_2^{(1)} & 0 & h_3^{(1)} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & h_2^{(1)} & 0 & h_3^{(1)} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & h_3^{(1)} \\
h_1^{(2)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & h_1^{(2)} & 0 & h_2^{(2)} & 0 & 0 & h_3^{(2)} & 0 & 0 \\
0 & 0 & h_1^{(2)} & 0 & h_2^{(2)} & 0 & 0 & h_3^{(2)} & 0 \\
0 & 0 & 0 & 0 & 0 & h_2^{(2)} & 0 & 0 & h_3^{(2)} \\
h_1^{(3)} & 0 & 0 & h_2^{(3)} & 0 & 0 & h_3^{(3)} & 0 & 0 \\
0 & h_1^{(3)} & 0 & 0 & h_2^{(3)} & 0 & 0 & h_3^{(3)} & 0 \\
0 & 0 & h_1^{(3)} & 0 & 0 & h_2^{(3)} & 0 & 0 & h_3^{(3)}
\end{bmatrix}
\tag{1.3}
$$

So the approach that was made to solve this problem was the same as the one we used to solve the 2-user problem. The probabilistic graphical model for the transmissions mentioned in 1.4 is the following:

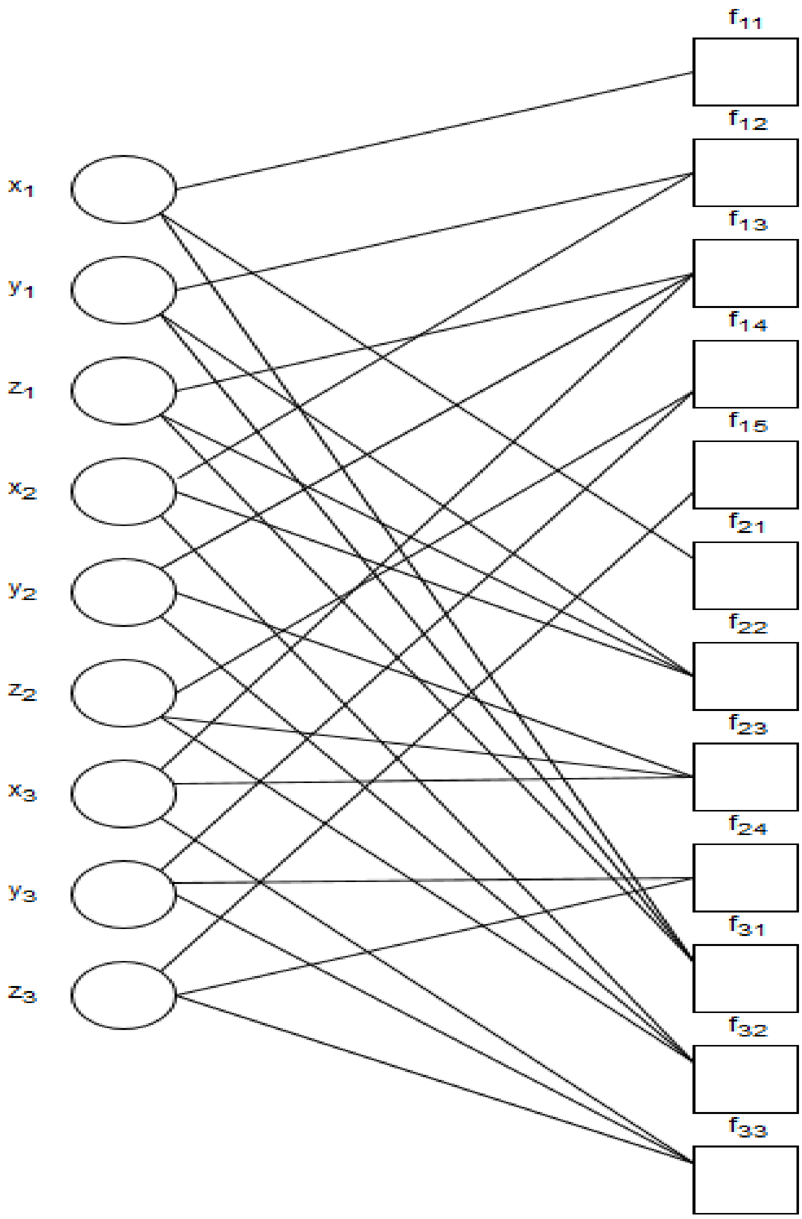

Figure 1.5: Factor graph based on Figure 2.4

After that the factor graph is simplified by applying the factor graph design equations that are mentioned in 2.1.3 and then proved in 6.1.

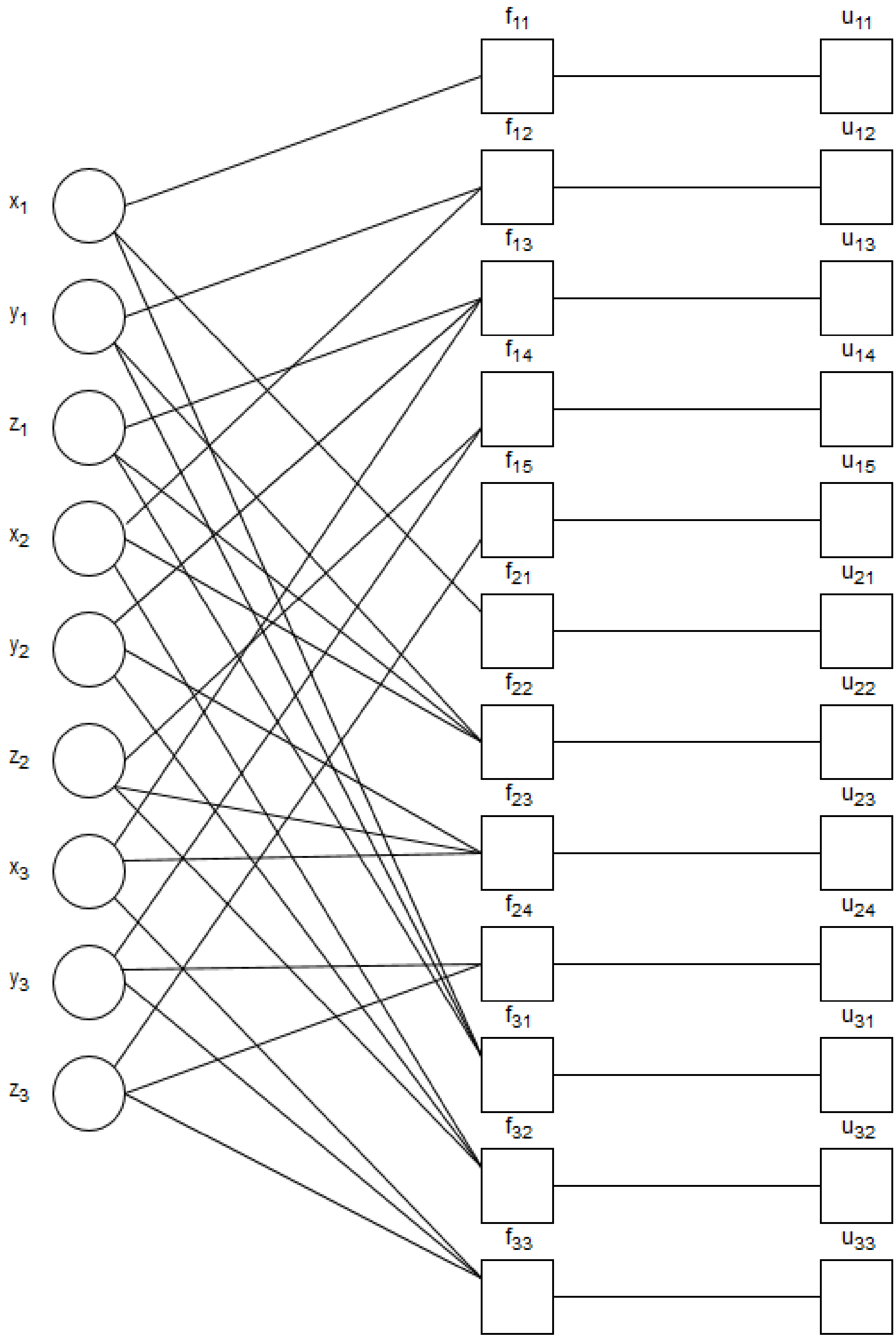

Figure 1.6: Factor graph based on 1.4 after the Factor Node Design

Chapter 2

Inference Algorithms

In this chapter, the Sum-Product Algorithm is explained based on the breakthrough paper [4]. Additionally descriptions are given on the factor graphs and on the factor node design that was used in 1.3.1 and 1.3.2. Lastly the stopping conditions that were used in our simulations are explained.

2.1 Sum-Product Algorithm in Factor Graphs

Definition of Factor Graphs [4]:

A *factor graph* is a bipartite graph that expresses the structure of the factorization. A *factor graph* has a variable node for each variable x_i, a *factor node* for each local function f_j, and an edge-connecting variable node x_i to factor node f_j if and only if x_i is an argument of f_j.

Thus a factor graph is a standard bipartite graphical representation of a mathematical relation.

2.1.1 Sum-Product Algorithm

The **Sum-Product message passing algorithm** also known as **belief propagation** is a message passing algorithm that performs inference on graphical models, such as Bayesian networks or Markov random fields. The basic idea of this algorithm is to compute the marginal distribution of the unobserved nodes, based on the conditional distribution of the observed nodes. There are many variations of the Sum-Product algorithm but this work revolves around the variation that operates on factor graphs.

2.1.2 Message passing

The aforementioned algorithms core idea is to pass real valued functions called messages along with the edges between the hidden nodes. More precisely, if v is a variable node and a is a factor node connected to v in the factor graph, the messages from v to a, (denoted by $\mu_{v\to a}$) and from a to v (denoted by $\mu_{a\to v}$) are real-valued functions. These messages contain the "influence" that one variable exerts on another. The messages are computed differently depending on whether the node receiving the message is a variable node or a factor node. Keeping the same notation:

- A message from a variable node v to a factor node a is the product of the messages from all other neighboring factor nodes :

$$\mu_{v\to a}(x_v) = \prod_{a^*\in N(v)\backslash a} \mu_{a^*\to v}(x_v). \tag{2.1}$$

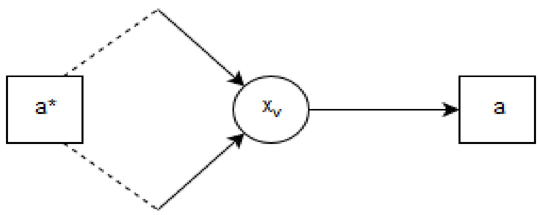

Figure 2.1: Message sent from variable node to factor node

where $N(v)$ is the set of neighboring (factor) nodes to v. If $N(v)\backslash a$ is empty, then $\mu_{v\to a}(x_v)$ is set to the uniform distribution.

- A message from a factor node a to a variable node v is the product of the factor with messages from all other nodes, marginalized over all variables except the one associated with v:

$$\mu_{a\to v}(x_v) = \sum_{x'_a : x'_v = x_v} f_a(x'_a) \prod_{v^*\in N(a)\backslash\{v\}} \mu_{v^*\to a}(x'_{v^*}) \tag{2.2}$$

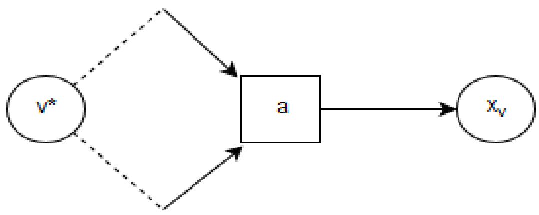

Figure 2.2: Message from factor node to variable node

where $N(a)$ is the set of neighboring (variable) nodes to a. If $N(a)\backslash v$ is empty then $\mu_{a\to v}(x_v) = f_a(x_v)$, since in this case $x_v = x_a$.

As shown by the previous formula: the complete marginalization is reduced to a sum of products of simpler terms than the ones appearing in the full joint distribution.This is the reason why it is called the sum-product algorithm.

In sum-product there are two different types of scheduling, the one being serial and the other one being parallel.In this work, the scheduling that was chosen was the parallel.

2.1.3 Factor Node Design

As it is mentioned in chapter 1.3 the factor nodes are initialized to simplify the factor graph. This simplification is achieved by applying the Bayes theorem in the factor node design equations. A full proof about the factor node design is given in 6.1 for N=2 and N=3 users.

2.2 Application of the Sum-Product in this work

In this section, two examples are given about how all the theory that was mentioned in 2.1 functions upon the cases we studied in chapters 1.3.1 and 1.3.2.

Proper Complex Gaussian Vector:

It is important to define proper complex Gaussian vector $\vec{r} \sim \mathcal{CN}(\vec{\mu}, \Lambda)$ where

- $\vec{r} \in \mathbb{C}^{\mathbb{N}}$
- $E[\vec{r}] = \vec{\mu}$
- $E[(\vec{r} - \vec{\mu})(\vec{r} - \vec{\mu})^H] = \Lambda$
- $f_{\vec{r}}(\vec{r}) = \frac{1}{\pi^N |\Lambda|} e^{-(\vec{r}-\vec{\mu})^H \Lambda^{-1} (\vec{r}-\vec{\mu})}$

For N=1 we get the circularly symmetric complex Gaussian

Circularly Symmetric Complex Gaussian:

- $\vec{r} \in \mathbb{C}$
- $E[\vec{r}] = \vec{0}$
- $E[\vec{r}\vec{r}^H] = \Lambda$
- $f_{\vec{r}}(\vec{r}) = \frac{1}{\pi |\Lambda|} e^{-\vec{r}^H \Lambda^{-1} \vec{r}}$

2.2.1 Application of the Sum-Product in the 2-User case

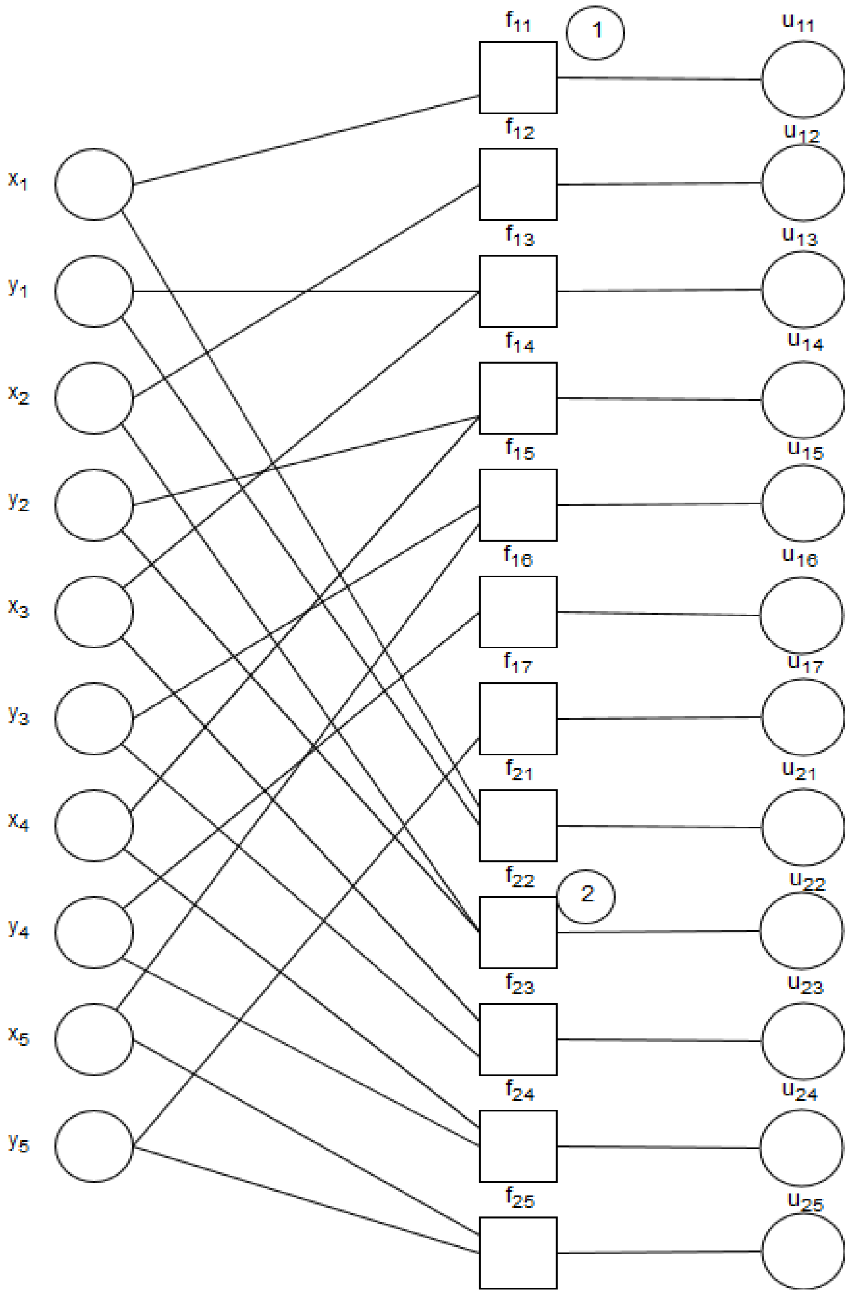

Figure 2.3: Example based on case 1 and 2 upon the factor graph

In this part, the application of the Sum-Product algorithm is presented based on the example that was covered in 1.3.1. To start with, an example that shows how the factor node design works (2.3) is presented. Observing 2.3 we can discriminate two different factor types, those with 1 edge and those with 2 edges. It is important to note that the factor nodes get designed based on the Circularly Symmetric Complex Gaussian.

- In case 1 (considering BPSK modulation) where we have just one edge, the factor node design is the following:

$$f_{11}(x_1 = +1) = \frac{\mathcal{CN}(u_{11}, h_1^{(1)}, \sigma^2)}{total}$$
$$f_{11}(x_1 = -1) = \frac{\mathcal{CN}(u_{11}, -h_1^{(1)}, \sigma^2)}{total}$$

- In case 2 (considering BPSK modulation) where we have two edges, the factor node design is the following:

$$f_{22}(x_2 = +1, y_2 = +1) = \frac{\mathcal{CN}(u_{22}, h_1^{(2)} + h_2^{(2)}, \sigma^2)}{total}$$
$$f_{22}(x_2 = +1, y_2 = -1) = \frac{\mathcal{CN}(u_{22}, h_1^{(2)} - h_2^{(2)}, \sigma^2)}{total}$$
$$f_{22}(x_2 = -1, y_2 = +1) = \frac{\mathcal{CN}(u_{22}, -h_1^{(2)} + h_2^{(2)}, \sigma^2)}{total}$$
$$f_{22}(x_2 = -1, y_2 = -1) = \frac{\mathcal{CN}(u_{22}, -h_1^{(2)} - h_2^{(2)}, \sigma^2)}{total}$$

After all the factors get designed the message passing phase begins. As it was mentioned in 2.1.2 there are two versions of sum-product, in this work the parallel version is being used. In the parallel sum-product there are 2 core steps:

Initialization: In the initialization step the messages from variables to factors got initialized to 0.5 each

Update: In the first sub-step of the update rule messages were sent from all the factor nodes to all the variable nodes (with edges in-between them) like it was mentioned in 2.2 In the second sub-step of the update rule messages were sent from all the variable nodes to all the factor nodes (with edges in-between them) like it was mentioned in 2.1

An example is given based on the case 2 of 2.3.

Factor to variable message:

$$\mu^{t}_{f_{22}\to y_2}(y_2 = -1) = f_{22}(x_2 = -1, y_2 = -1)\mu^{t-1}_{x_2\to f_{22}}(x_2 = -1)+$$
$$f_{22}(x_2 = +1, y_2 = -1)\mu^{t-1}_{x_2\to f_{22}}(x_2 = +1)$$
$$\mu^{t}_{f_{22}\to y_2}(y_2 = +1) = f_{22}(x_2 = -1, y_2 = +1)\mu^{t-1}_{x_2\to f_{22}}(x_2 = -1)+$$
$$f_{22}(x_2 = +1, y_2 = +1)\mu^{t-1}_{x_2\to f_{22}}(x_2 = +1)$$

The binary message sent is the message from factor f_{22} to variable y_2. It is important to note that before we sent it we have to normalize it.

Variable to factor message:

$$\mu^{t}_{y_2\to f_{22}}(y_2) = \mu^{t-1}_{f_{14}\to y_2}(y_2)$$

The message passing continues until the stopping condition. The stopping condition that was used in this work was a heuristic one where the algorithm compares the messages it sent on the $t - 1$ iteration with the messages it sent on the t iteration and if they did not change more than a threshold it terminates.

If the threshold is a large number, that means that the tolerance for terminating the algorithm is large which means that it will do less iterations with the trade-off being slightly higher bit error rate. On the other hand if the threshold is a small number that means that the algorithm will have smaller bit error rate with the trade-off being more iterations.

2.2.2 Application of the Sum-Product in the 3-User case

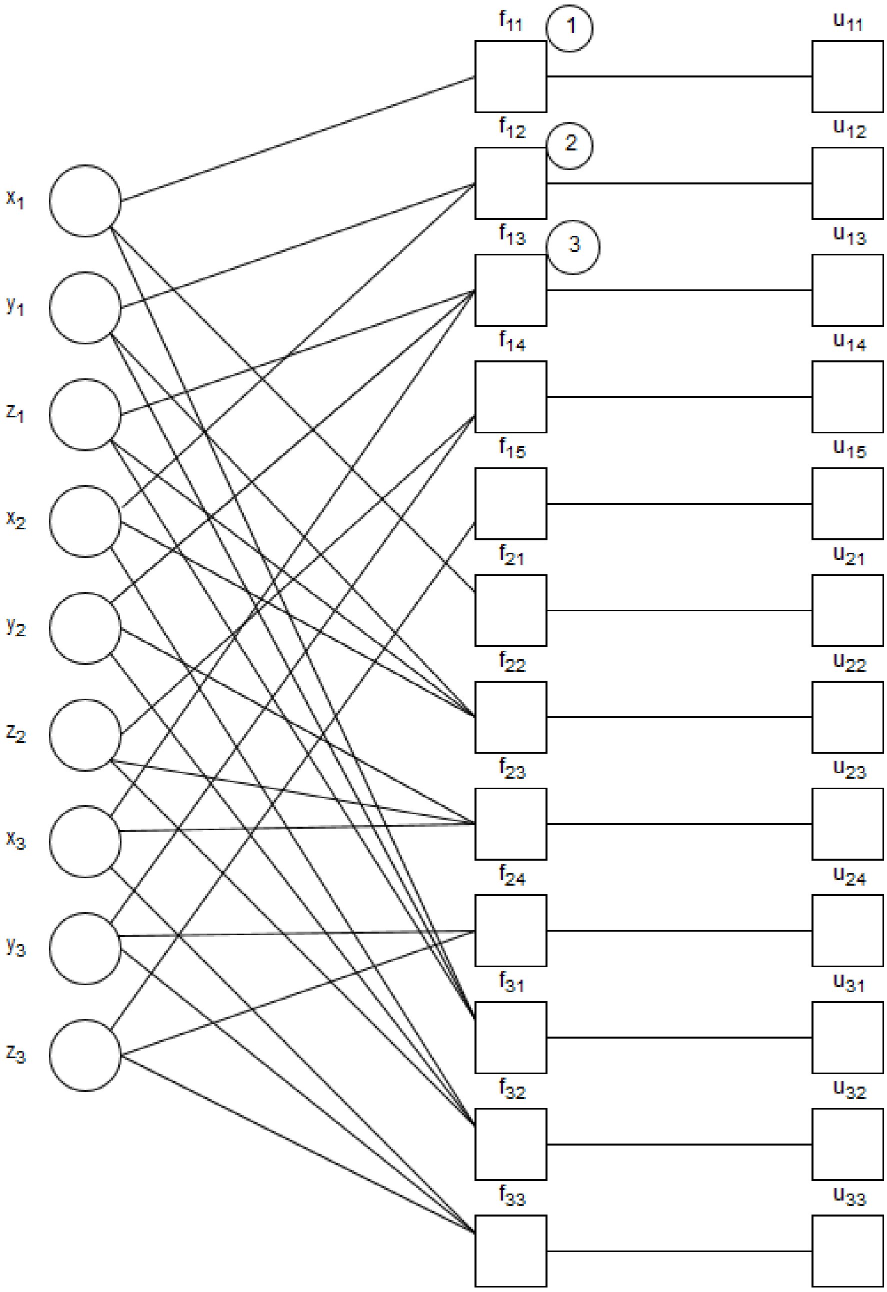

Figure 2.4: Example based on case 1,2 and 3 upon the factor graph

In this part, the application of the Sum-Product algorithm is presented based on the example that was covered in 1.3.1. To start with, an example that shows how the factor node initialization works (2.4) is presented. Observing 2.4 we can discriminate three different factor types, those with 1 edge, those with 2 edges and those with 3 edges. It is important to note that the factor nodes get designed based on the Circularly Symmetric Complex Gaussian.

- In case 1 (considering BPSK modulation) where we have just one edge, the factor node design is the following:

$$f_{11}(x_1 = +1) = \frac{\mathcal{CN}(u_{11}, h_1^{(1)}, \sigma^2)}{total}$$
$$f_{11}(x_1 = -1) = \frac{\mathcal{CN}(u_{11}, -h_1^{(1)}, \sigma^2)}{total}$$

- In case 2 (considering BPSK modulation) where we have two edges, the factor node design is the following:

$$f_{12}(x_2 = +1, y_1 = +1) = \frac{\mathcal{CN}(u_{12}, h_1^{(1)} + h_2^{(1)}, \sigma^2)}{total}$$
$$f_{12}(x_2 = +1, y_1 = -1) = \frac{\mathcal{CN}(u_{12}, h_1^{(1)} - h_2^{(1)}, \sigma^2)}{total}$$
$$f_{12}(x_2 = -1, y_1 = +1) = \frac{\mathcal{CN}(u_{12}, -h_1^{(1)} + h_2^{(1)}, \sigma^2)}{total}$$
$$f_{12}(x_2 = -1, y_1 = -1) = \frac{\mathcal{CN}(u_{12}, -h_1^{(1)} - h_2^{(1)}, \sigma^2)}{total}$$

- In case 3(considering BPSK modulation) where we have three edges, the factor node design is the following:

$$f_{13}(x_3 = +1, y_2 = +1, z_1 = +1) = \frac{\mathcal{CN}(u_{13}, h_1^{(1)} + h_2^{(1)} + h_3^{(1)}, \sigma^2)}{total}$$
$$f_{13}(x_3 = +1, y_2 = +1, z_1 = -1) = \frac{\mathcal{CN}(u_{13}, h_1^{(1)} + h_2^{(1)} - h_3^{(1)}, \sigma^2)}{total}$$
$$f_{13}(x_3 = +1, y_2 = -1, z_1 = +1) = \frac{\mathcal{CN}(u_{13}, h_1^{(1)} - h_2^{(1)} + h_3^{(1)}, \sigma^2)}{total}$$
$$f_{13}(x_3 = +1, y_2 = -1, z_1 = -1) = \frac{\mathcal{CN}(u_{13}, h_1^{(1)} - h_2^{(1)} - h_3^{(1)}, \sigma^2)}{total}$$
$$f_{13}(x_3 = -1, y_2 = +1, z_1 = +1) = \frac{\mathcal{CN}(u_{13}, -h_1^{(1)} + h_2^{(1)} + h_3^{(1)}, \sigma^2)}{total}$$
$$f_{13}(x_3 = -1, y_2 = +1, z_1 = -1) = \frac{\mathcal{CN}(u_{13}, -h_1^{(1)} + h_2^{(1)} - h_3^{(1)}, \sigma^2)}{total}$$
$$f_{13}(x_3 = -1, y_2 = -1, z_1 = +1) = \frac{\mathcal{CN}(u_{13}, -h_1^{(1)} - h_2^{(1)} + h_3^{(1)}, \sigma^2)}{total}$$
$$f_{13}(x_3 = -1, y_2 = -1, z_1 = -1) = \frac{\mathcal{CN}(u_{13}, -h_1^{(1)} - h_2^{(1)} - h_3^{(1)}, \sigma^2)}{total}$$

After all the factors get initialized the message passing phase begins. As it was mentioned in 2.1.2 there are two versions of sum-product, in this work the parallel version is being used. In the parallel sum-product there are 2 core steps:

Initialization: In the initialization step the messages from variables to factors got initialized to 0.5 each

Update: In the first sub-step of the update rule messages were sent from all the factor nodes to all the variable nodes (with edges in-between them) like it was mentioned in 2.2 In the second sub-step of the update rule messages were sent from all the variable nodes to all the factor nodes (with edges in-between them) like it was mentioned in 2.1

An example is given based on the case 3 of 2.4.

Factor to variable message:

$$\begin{aligned}
&\mu^{t}_{f_{13}\to z_1}(z_1=-1)=\\
&f_{13}(x_3=+1,y_2=+1,z_1=-1)\mu^{t-1}_{x_3\to f_{13}}(x_3=+1)\mu^{t-1}_{y_2\to f_{13}}(y_2=+1)+\\
&f_{13}(x_3=+1,y_2=-1,z_1=-1)\mu^{t-1}_{x_3\to f_{13}}(x_3=+1)\mu^{t-1}_{y_2\to f_{13}}(y_2=-1)+\\
&f_{13}(x_3=-1,y_2=+1,z_1=-1)\mu^{t-1}_{x_3\to f_{13}}(x_3=-1)\mu^{t-1}_{y_2\to f_{13}}(y_2=+1)+\\
&f_{13}(x_3=-1,y_2=-1,z_1=-1)\mu^{t-1}_{x_3\to f_{13}}(x_3=-1)\mu^{t-1}_{y_2\to f_{13}}(y_2=-1)
\end{aligned}$$

$$\begin{aligned}
&\mu^{t}_{f_{13}\to z_1}(z_1=+1)=\\
&f_{13}(x_3=+1,y_2=+1,z_1=+1)\mu^{t-1}_{x_3\to f_{13}}(x_3=+1)\mu^{t-1}_{y_2\to f_{13}}(y_2=+1)+\\
&f_{13}(x_3=+1,y_2=-1,z_1=+1)\mu^{t-1}_{x_3\to f_{13}}(x_3=+1)\mu^{t-1}_{y_2\to f_{13}}(y_2=-1)+\\
&f_{13}(x_3=-1,y_2=+1,z_1=+1)\mu^{t-1}_{x_3\to f_{13}}(x_3=-1)\mu^{t-1}_{y_2\to f_{13}}(y_2=+1)+\\
&f_{13}(x_3=-1,y_2=-1,z_1=+1)\mu^{t-1}_{x_3\to f_{13}}(x_3=-1)\mu^{t-1}_{y_2\to f_{13}}(y_2=-1)
\end{aligned}$$

The binary message sent is the message from factor f_{13} to variable z_1. It is important to note that before we sent it we have to normalize it.

Variable to factor message:

$$\mu^{t}_{z_1\to f_{13}}(z_1)=\mu^{t-1}_{f_{22}\to z_1}(z_1)\mu^{t-1}_{f_{31}\to z_1}(z_1)$$

As it was mentioned before, the message passing continues until the stopping condition. The stopping condition that was used in this work was a heuristic one where the algorithm compares the messages it sent on the $t-1$ iteration with the messages it sent on the t iteration and if they did not change more than a threshold it terminates.

If the threshold is a large number, that means that the tolerance for terminating the algorithm is large which means that it will do less iterations with the trade-off being slightly higher bit error rate. On the other hand if the threshold is a small number that means that the algorithm will have smaller bit error rate with the trade-off being more iterations.

Chapter 3

Channel Estimation

The system models in 1.3 assumes CSI knowledge, thus for completeness sake it is important to estimate the channels of each user.

In this chapter, it is assumed that each user sends a preamble with his data packet that is known at the access point. In 1.3, it is also assumed that the time delay of the 2nd user compared to the 1st user is known on the access point. Thus with the knowledge of these two things, the LMMSE estimator is applied on the problem so that the channel can be estimated.

3.1 LMMSE estimator

The LMMSE estimator definition and the Bayesian Gauss Markov Theorem that are described, are based on [5].

It is assumed that there is a parameter $\vec{\theta}$ that needs to be estimated based on the data set $\{q[0], q[1], ..., q[N-1]\}$ or in vector form $\vec{q} = [q[0]\ q[1]\ ...\ q[N-1]]^T$. The unknown parameter is modelled as the realization of a random variable. There is no need for knowledge of the joint PDF $p(\vec{q}, \vec{\theta})$, however it is necessary to know the first two moments. $\vec{\theta}$ can be estimated from $\vec{q}$ due to the assumed statistical dependence of $\vec{\theta}$ on $\vec{q}$ as summarized by the joint PDF $p(\vec{q}, \vec{\theta})$ and in particular, for a linear estimator we rely on the correlation between $\vec{\theta}$ and $\vec{q}$. The class of all linear estimators are considered to have the following form:

$$\hat{\vec{\theta}} = \sum_{n=0}^{N-1} a_n q[n] + b_N \tag{3.1}$$

and the choice of the weighting coefficients a_n's so that the Bayesian MSE gets minimized are:

$$Bmse(\hat{\vec{\theta}}) = E[(\vec{\theta} - \hat{\vec{\theta}})^2] \tag{3.2}$$

where the expectation is with respect to the PDF $p(\vec{q}, \vec{\theta})$. The estimator that was described above is the **linear minimum mean square error (LMMSE) estimator**.

Bayesian Gauss-Markov Theorem If the data are described by the Bayesian linear model form

$$q = A\theta + w \tag{3.3}$$

where $\vec{q}$ is an $N \times 1$ data vector, A is a known $N \times p$ observation matrix, $\vec{\theta}$ is a $p \times 1$ random vector of parameters whose realization is to be estimated and has mean $\mathrm{E}(\vec{\theta})$ and covariance matrix $C_{\theta\theta}$, and w is an $N \times 1$ random vector with zero mean and covariance matrix C_w and is uncorrelated with $\vec{\theta}$, then the LMMSE estimator of $\vec{theta}$ is:

$$\hat{\theta} = E(\vec{\theta}) + (C_{\theta\theta}^{-1} + A^T C_{ww}^{-1} A)^{-1} A^T C_w^{-1} (q - AE(\vec{\theta})) \tag{3.4}$$

3.2 LMMSE application

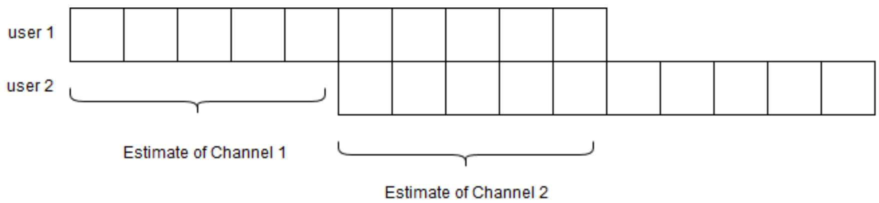

Figure 3.1: Application of Bayesian Markov Theorem on the preamble of the 2-User system model for channel estimation

In this part, the Bayesian Gauss Markov Theorem is applied on the problem that was mentioned in 1.3.1. Considering that the access point knows the preamble and the time delay as it was mentioned before, by applying the Bayesian Gauss Markov Theorem on the preamble of the 2 users, their channels can be estimated as it is shown in 3.1. It is worth noting that Bayesian Gauss Markov Theorem generally works for the real case, and the reason that it can be applied here is that As in the example of 3.1, the access point receives the linear equations for the 5 first slots:

$$\vec{u} = \vec{x} h_1 + \vec{v} \tag{3.5}$$

which are the same as the 3.3, so we can apply the Bayesian Gauss Markov Theorem. x are the known preamble bits, h_1 is the channel that needs to be estimated and v is the noise. So according to the 3.4

$$\hat{h_1} = E(h_1) + (C_{h_1h_1}^{-1} + x^T C_{vv}^{-1} x)^{-1} x^T C_v^{-1}(u - xE(h_1)) \tag{3.6}$$

- $E[h_1] = 0$ because Rayleigh Fading was assumed
- $C_{h_1h_1}^{-1} = var(h_1) = 1$
- $C_{vv}^{-1} = E[ww^H] = \sigma^2 I$ where I is the identity matrix and sigma squared is the variance of the noise

so for the example that is mentioned in 3.1 the channel estimate is the following:

$$\hat{h_1} = (1 + [1\ 1\ 1\ 1\ 1]^T \sigma^2 I [1\ 1\ 1\ 1\ 1])[1\ 1\ 1\ 1\ 1]^T \sigma^2 I u \tag{3.7}$$

$$\hat{h_1} = (1 + [1\ 1\ 1\ 1\ 1]^T \sigma^2 I [1\ 1\ 1\ 1\ 1])[1\ 1\ 1\ 1\ 1]^T \sigma^2 I u \tag{3.8}$$

where sigma squared is the variance of the noise. After the 1st channel h_1 gets estimated, its estimate is placed on the equation for the 5 next slots:

$$\vec{u} = \vec{x}h_1 + \vec{x}h_2 + \vec{v} \tag{3.9}$$

and by replacing the xh_1 which is now known:

$$\vec{u} - \vec{x}h_1 = \vec{x}h_2 + \vec{v} \tag{3.10}$$

so by setting $\vec{u} - \vec{x}h_1 = \vec{u'}$ the above equation becomes like the one mentioned in 3.5. So by solving that equation with the exact same way we get an estimate for the channel h_2. It is worth noting that Bayesian Gauss Markov Theorem generally works for the real case, and the reason that it can be applied here is that the modulation that was used was BPSK so $C_{\theta\theta} = real$ and $C_{ww} = \sigma^2 I = real$. Thus Bayesian Gauss Markov Theorem can be applied.

That concludes the chapter of channel estimation and in the section 4 there are plots that show the estimation error of the method that is being used.

Chapter 4

Numerical Results

In this chapter, plots are provided for everything that was presented in this work. To start with, there are plots for the Channel Estimation that showcase the performance of the LMMSE over SNR. To continue with, there are plots that compare the performance of 3 Users to the performance of the 2 Users over packet length=100 Bits. Lastly there are plots that show the general performance of the algorithm of this work in different SNR's over different packet lengths.

4.1 Channel Estimation Plot

In this section, plots are presented based on the example mentioned in 3.1.

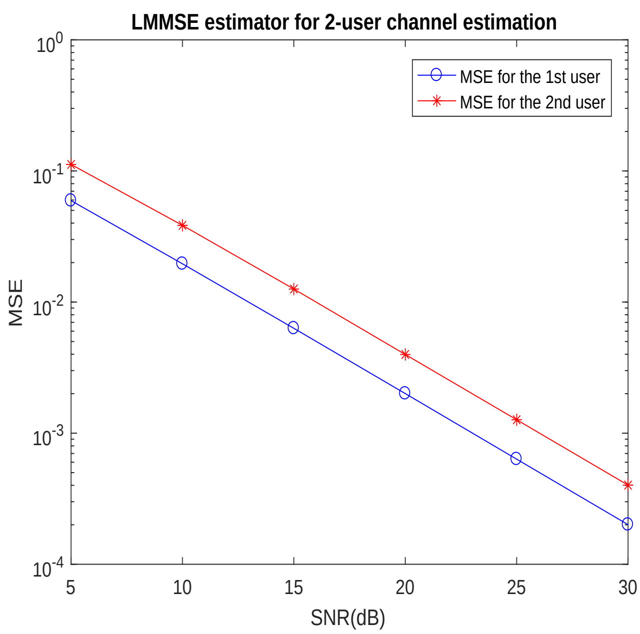

Figure 4.1: Performance of the LMMSE estimator on the system model of this work for N=2,3 users

The average MSE for the channel estimation of the 2nd user is higher than the one of the 1st due to the fact that the 2nd user's channel estimation considers in its calculation the 1st user's estimation.

4.2 BER/SNR plots for N=2 and N=3 users

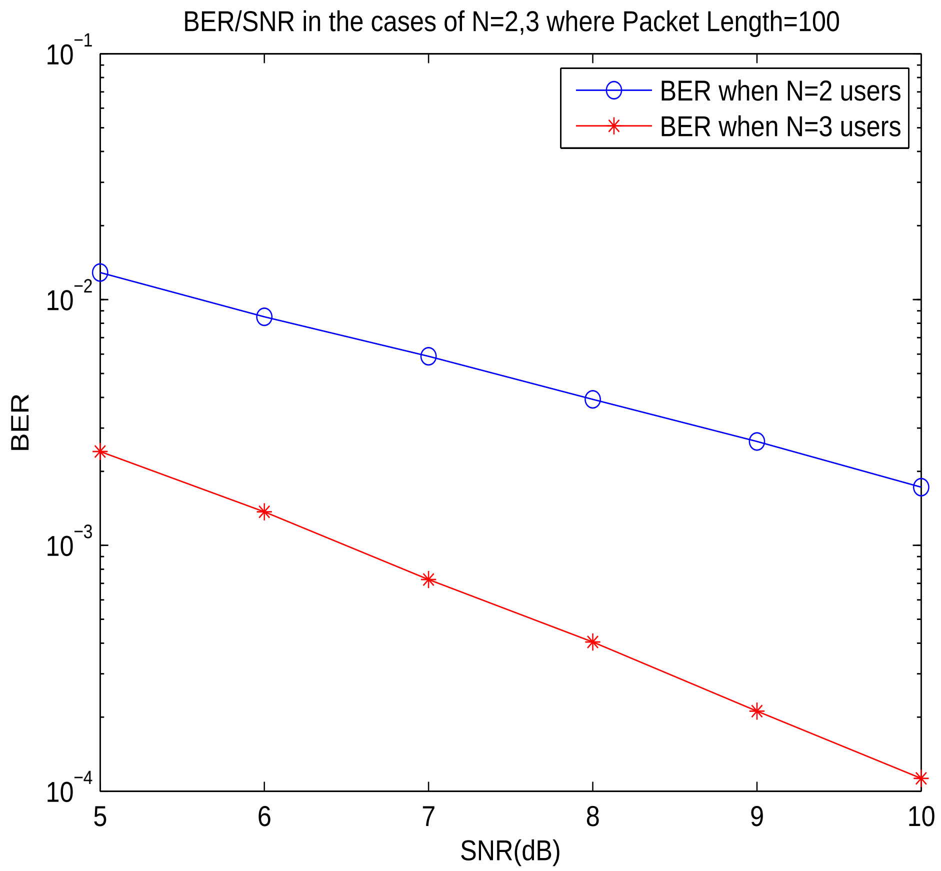

Figure 4.2: Comparison between the 2-user and the 3-user system models where packet length is B=100 bits

In this section, there is a comparison between the 2-user and the 3 user system models considering a fixed packet length of B=100 bits where the results point that the 3-user system model has a much smaller BER. The 3-user model seems to have about a 4 dB gain.

4.3 BER/Packet length plots for N=2 and N=3 users

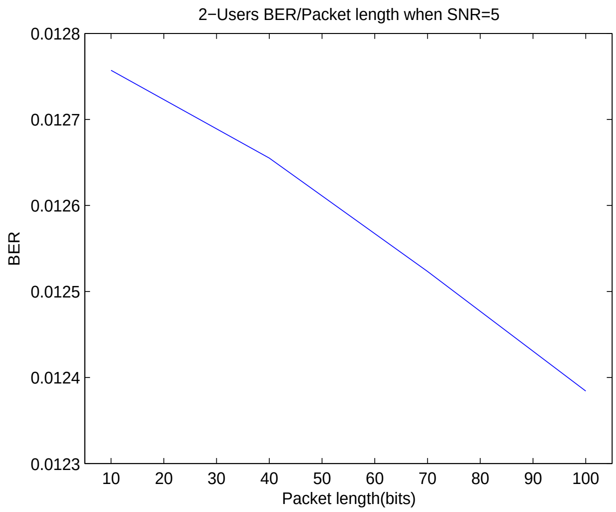

Figure 4.3: 2-User BER/Packet length plots over 5dB SNR

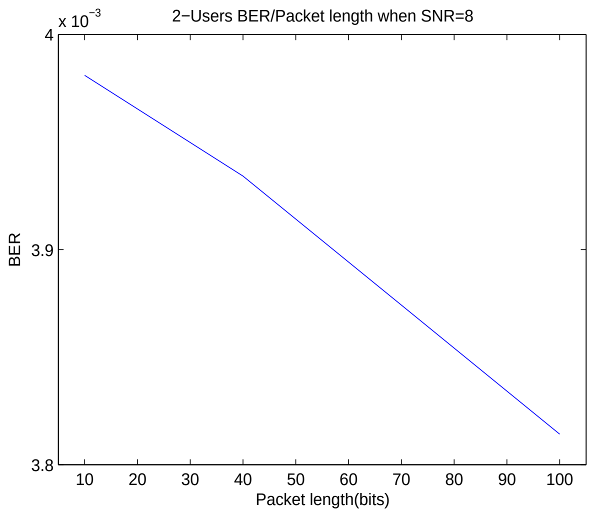

Figure 4.4: 2-User BER/Packet length plots over 8dB SNR

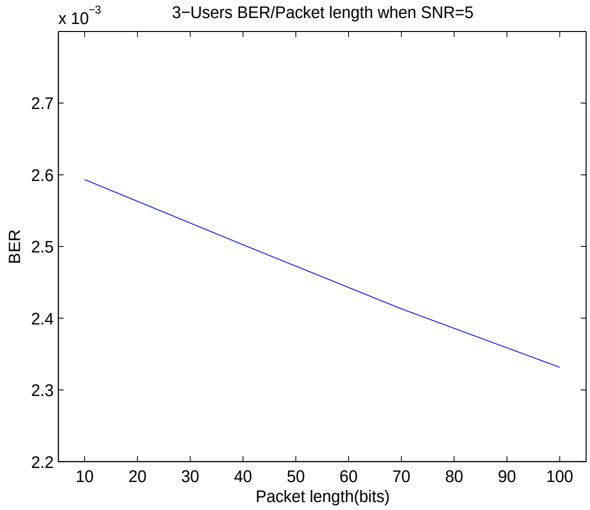

Figure 4.5: 3-User BER/Packet length plots over 5dB SNR

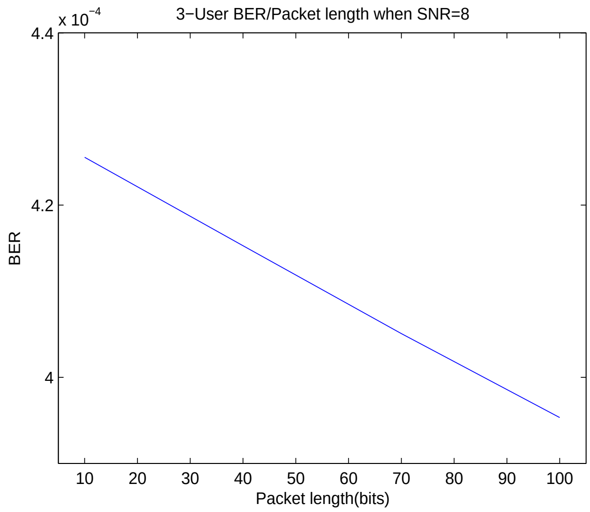

Figure 4.6: 3-User BER/Packet length plots over 8dB SNR

In these plots, it is shown that as the length of the packets grow the performance of the algorithm improves. That is due to the fact that the factor graph becomes more and more tree-like as the length of the packet grows and as it is known the sum-product algorithm is optimal for trees. This matches well with the results of [3] reference where they prove that as the length of the packets grow their algorithm becomes more optimal.

Chapter 5

Conclusion and future work

5.1 Conclusion

This work showcased state-of-the-art probabilistic graphical model and inference algorithm knowledge that can exploit the symbol level asynchrony in multi-user wireless communications.With many steps of implementation consisting of: Crafting a PGM, initializing its factor graph and providing message passing equations on it, this work proves that performance is sensitive to initialization and that a trade-off exists between convergence time and BER.

5.2 Future Work

In this work, there were assumptions that were not veritable such as the symbol-level asynchrony, the knowledge of the time delays in the access point and the bit durations that were considered equal at all times. An idea for the future would be, how would a system model that transcends these assumptions perform compared to the performance of the system model that was presented in this work. Also another idea for future work would be, channel estimation with the use of the data collected by the AP using an inference clustering algorithm such as affinity propagation. Lastly another idea would be to do joint CSI estimation instead of the sequential that was done in this work.

Chapter 6

Appendix

6.1 Factor Node Design Proof for N=2,3 Users

In this section the proof for the factor node design is provided for N=2 and N=3 users.

6.1.1 2-User Case Factor Node Design

Considering that c is the transmission round (as it was mentioned before 2-users means 2 transmissions), k is the k^{th} slot of the transmission round, x_i is the i^{th} transmitted symbol of the 1st user and y_j is the j^{th} transmitted symbol of the 2nd user the factor node equation is the following:

$$f_{ck}(x_i, y_j) = Pr(x_i, y_j | h_1^{(c)}, h_2^{(c)}, u_{ck}) \tag{6.1}$$

Bayes theorem notes that the following is true:

$$Pr(A|B,C) = \frac{Pr(A,C|B)}{Pr(C|B)} = \frac{Pr(C|A,B)Pr(A|B)}{\sum\limits_{A} Pr(A,C|B)} \tag{6.2}$$

Setting A=x_i, y_j B=$h_1^{(c)}, h_2^{(c)}$ and C=u_{ck}, and combining them with the aforementioned equations we get the following proof:

Proof.

$$f_{ck}(x_i,y_j) = Pr(x_i,y_j|h_1^{(c)},h_2^{(c)},u_{ck}) = \frac{Pr(x_i,y_j,u_{ck}|h_1^{(c)},h_2^{(c)})}{Pr(u_{ck}|h_1^{(c)},h_2^{(c)})} =$$

$$\frac{Pr(u_{ck}|h_1^{(c)},h_2^{(c)},x_i,y_j)Pr(x_i,y_j|h_1^{(c)},h_2^{(c)})}{\sum\limits_{x_i}\sum\limits_{y_j} Pr(u_{ck}|h_1^{(c)},h_2^{(c)},x_i,y_j)Pr(x_i)Pr(y_j)} =$$

$$\frac{Pr(u_{ck}|h_1^{(c)},h_2^{(c)},x_i,y_j)Pr(x_i)Pr(y_j)}{\sum\limits_{x_i}\sum\limits_{y_j} Pr(u_{ck}|h_1^{(c)},h_2^{(c)},x_i,y_j)Pr(x_i)Pr(y_j)} =$$

$$\frac{Pr(u_{ck}|h_1^{(c)},h_2^{(c)},x_i,y_j)}{\sum\limits_{x_i}\sum\limits_{y_j} Pr(u_{ck}|h_1^{(c)},h_2^{(c)},x_i,y_j)} = \frac{\mathcal{CN}(u_{ck},x_ih_1^{(c)}+y_jh_2^{(c)},\sigma^2)}{\sum\limits_{x_i}\sum\limits_{y_j}\mathcal{CN}(u_{ck},x_ih_1^{(c)}+y_jh_2^{(c)},\sigma^2)}$$

$\square$

Because $\bar{u}_{ck}$ is known in the AP, the factor node f_{ck} can be designed based on the above proof with the following equations:

$$f_{ck}(x_i=+1,y_j=+1) = \frac{\mathcal{CN}(u_{ck},h_1^{(c)}+h_2^{(c)},\sigma^2)}{total}$$
$$f_{ck}(x_i=+1,y_j=-1) = \frac{\mathcal{CN}(u_{ck},h_1^{(c)}-h_2^{(c)},\sigma^2)}{total}$$
$$f_{ck}(x_i=-1,y_j=+1) = \frac{\mathcal{CN}(u_{ck},-h_1^{(c)}+h_2^{(c)},\sigma^2)}{total}$$
$$f_{ck}(x_i=-1,y_j=-1) = \frac{\mathcal{CN}(u_{ck},-h_1^{(c)}-h_2^{(c)},\sigma^2)}{total}$$

where $\mathcal{CN}(u_{ck},h_1^{(c)}+h_2^{(c)},\sigma^2)$ stands for a complex normal probability density function with mean $h_1^{(c)}+h_2^{(c)}$(or any of the other 4 combinations), variance σ^2 evaluated at the observed value u_{ck}. Total is equal to the sum of all the possible factor node combinations.

6.1.2 3-User Case Factor Node Design

Considering that c is the transmission round(as it was mentioned before 2-users means 2 transmissions), k is the k^{th} slot of the transmission round, x_i is the i^{th} transmitted symbol of the 1st user, y_j is the j^{th} transmitted symbol of the 2nd user and z_v is the v^{th} transmitted symbol of the 3d user the factor node equation is the following:

$$f_{ck}(x_i, y_j, z_v) = Pr(x_i, y_j, z_v | h_1^{(c)}, h_2^{(c)}, h_3^{(c)}, u_{ck}) \tag{6.3}$$

Bayes theorem notes that the following is true:

$$Pr(A|B,C) = \frac{Pr(A,C|B)}{Pr(C|B)} = \frac{Pr(C|A,B)Pr(A|B)}{\sum\limits_{A} Pr(A,C|B)} \tag{6.4}$$

Setting A=x_i, y_j, z_v B=$h_1^{(c)}, h_2^{(c)}, h_3^{(c)}$ and C=u_{ck} ,and combining them with the aforementioned equations we get the following proof:

Proof.

$$f_{ck}(x_i, y_j, z_v) = Pr(x_i, y_j, z_v | h_1^{(c)}, h_2^{(c)}, h_3^{(c)}, \bar{u}_{ck}) = \frac{Pr(x_i, y_j, z_v, u_{ck} | h_1^{(c)}, h_2^{(c)}, h_3^{(c)})}{Pr(u_{ck} | h_1^{(c)}, h_2^{(c)}, h_3^{(c)})} =$$

$$\frac{Pr(u_{ck} | h_1^{(c)}, h_2^{(c)}, h_3^{(c)}, x_i, y_j, z_v) Pr(x_i, y_j, z_v | h_1^{(c)}, h_2^{(c)}, h_3^{(c)})}{\sum\limits_{x_i}\sum\limits_{y_j}\sum\limits_{z_v} Pr(u_{ck} | h_1^{(c)}, h_2^{(c)}, h_3^{(c)}, x_i, y_j, z_v) Pr(x_i) Pr(y_j) Pr(z_v)} =$$

$$\frac{Pr(u_{ck} | h_1^{(c)}, h_2^{(c)}, h_3^{(c)}, x_i, y_j, z_v) Pr(x_i) Pr(y_j) Pr(z_v)}{\sum\limits_{x_i}\sum\limits_{y_j}\sum\limits_{z_v} Pr(u_{ck} | h_1^{(c)}, h_2^{(c)}, h_3^{(c)}, x_i, y_j, z_v) Pr(x_i) Pr(y_j) Pr(z_v)} =$$

$$\frac{Pr(u_{ck} | h_1^{(c)}, h_2^{(c)}, h_3^{(c)} x_i, y_j, z_v)}{\sum\limits_{x_i}\sum\limits_{y_j}\sum\limits_{z_v} Pr(u_{ck} | h_1^{(c)}, h_2^{(c)}, h_3^{(c)}, x_i, y_j, z_v)} = \frac{\mathcal{CN}(u_{ck}, x_i h_1^{(c)} + y_j h_2^{(c)} + z_v h_3^{(c)}, \sigma^2)}{\sum\limits_{x_i}\sum\limits_{y_j}\sum\limits_{z_v} \mathcal{CN}(u_{ck}, x_i h_1^{(c)} + y_j h_2^{(c)} + z_v h_3^{(c)}, \sigma^2)}$$

□

Because u_{ck} is known in the AP, the factor node f_{ck} can be designed based on the above proof with the following equations:

$$f_{ck}(x_i = +1, y_j = +1, z_v = +1) = \frac{\mathcal{CN}(u_{ck}, h_1^{(c)} + h_2^{(c)} + h_3^{(c)}, \sigma^2)}{total}$$
$$f_{ck}(x_i = +1, y_j = +1, z_v = -1) = \frac{\mathcal{CN}(u_{ck}, h_1^{(c)} + h_2^{(c)} - h_3^{(c)}, \sigma^2)}{total}$$
$$f_{ck}(x_i = +1, y_j = -1, z_v = +1) = \frac{\mathcal{CN}(u_{ck}, h_1^{(c)} - h_2^{(c)} + h_3^{(c)}, \sigma^2)}{total}$$
$$f_{ck}(x_i = +1, y_j = -1, z_v = -1) = \frac{\mathcal{CN}(u_{ck}, h_1^{(c)} - h_2^{(c)} - h_3^{(c)}, \sigma^2)}{total}$$
$$f_{ck}(x_i = -1, y_j = +1, z_v = +1) = \frac{\mathcal{CN}(u_{ck}, -h_1^{(c)} + h_2^{(c)} + h_3^{(c)}, \sigma^2)}{total}$$
$$f_{ck}(x_i = -1, y_j = +1, z_v = -1) = \frac{\mathcal{CN}(u_{ck}, -h_1^{(c)} + h_2^{(c)} - h_3^{(c)}, \sigma^2)}{total}$$
$$f_{ck}(x_i = -1, y_j = -1, z_v = +1) = \frac{\mathcal{CN}(u_{ck}, -h_1^{(c)} - h_2^{(c)} + h_3^{(c)}, \sigma^2)}{total}$$
$$f_{ck}(x_i = -1, y_j = -1, z_v = -1) = \frac{\mathcal{CN}(u_{ck}, -h_1^{(c)} - h_2^{(c)} - h_3^{(c)}, \sigma^2)}{total}$$

where $\mathcal{CN}(u_{ck}, h_1^{(c)} + h_2^{(c)} + h_3^{(c)}, \sigma^2)$ stands for a complex normal probability density function with mean $h_1^{(c)} + h_2^{(c)} + h_3^{(c)}$(or any of the other 8 combinations), variance σ^2 evaluated at the observed value u_{ck}. Total is equal to the sum of all the possible factor node combinations.

Bibliography

[1] J. Boutros and G. Caire, "Iterative multiuser joint decoding: unified framework and asymptotic analysis," *IEEE Transactions on Information Theory*, vol. 48, no. 7, pp. 1772–1793, Jul 2002.

[2] S. Gollakota and D. Katabi, "Zigzag decoding: Combating hidden terminals in wireless networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 159–170, Aug. 2008. [Online]. Available: http://doi.acm.org/10.1145/1402946.1402977

[3] A. S. Tehrani, A. G. Dimakis, and M. J. Neely, "Sigsag: Iterative detection through soft message-passing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 8, pp. 1512–1523, Dec 2011.

[4] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, Feb 2001.

[5] S. M. Kay, *Fundamentals of statistical signal processing*, ser. Prentice Hall signal processing series. Upper Saddle River, NJ: Prentice Hall PTR, 1993. [Online]. Available: https://cds.cern.ch/record/2012069