TECHNICAL UNIVERSITY OF CRETE

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

Asynchronous Inference for Security Applications

By Zacharias Iosif Sifakis

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

ELECTRICAL AND COMPUTER ENGINEERING

October 2022

THESIS COMMITTEE

Professor Aggelos Bletsas, *Thesis Supervisor*

Professor George N. Karystinos

Professor Michail G. Lagoudakis

ABSTRACT

This work studies distributed inference for privacy-non intrusive security applications. Specifically, minimum mean square estimation (MMSE) of channel parameters is proposed, with asynchronous and distributed implementation of Gaussian Belief Propagation (GBP). Wireless channel is measured by distributed wireless terminals/sensors, which exchange messages to infer the wireless channel through asynchronous GBP. Variations of the channel between consecutive measurements indicate the presence (or absence) of a moving person, without other means (e.g., cameras, microphones or infrared sensors). The method is tested in simulations and compared to synchronous GBP in terms of convergence; in addition, the method is contrasted against centralized least-squares (LS) and centralized MMSE, at the WiFi carrier frequency of 2.4 GHz, at various signal-to-noise ratios. It is found that mobility changes channel estimate metrics by at least one order of magnitude, compared to the static (immobile) case. The method could be exploited in indoor environments, where distributed presence of embedded wireless radios is widespread, without reverting to privacy intrusive microphones or cameras. In addition, the method is truly asynchronous and distributed, and hence more robust compared to synchronous counterparts.

ACKNOWLEDGEMENTS

First and foremost, I thank my Thesis Supervisor, Prof. Aggelos Bletsas, for his tremendous support and guidance throughout this entire process. I also thank the members of the commitee of this thesis, Professors George Karystinos and Michail Lagoudakis for evaluating my work. I am also grateful to the professors I've had all those years.

Furthermore, I would like to thank the graduate students Vassilis Papageorgiou and George Apostolakis for their assistance. I would also like to thank my fellow students and friends for their help and support. Last, but not least, I thank my family for being present.

-Zacharias Iosif Sifakis, October 2022

Contents

Chapter 1

Introduction

A major issue in a plethora of problems is inference based on a set of measurements from a number of agents on a distributed network. Although centralized algorithms can be used on a small-scale network, they face challenges in large-scale networks, placing a heavy burden on communication when all data has to be transported and processed to a central processing unit. As a result, distributed inference techniques that rely strictly on local communication and computation are vital for problems occurring within distributed networks. [1]

The use of message-passing algorithms (e.g. sum-product, max-product), also known as belief propagation has provided detailed examples of how decision-making and inference can be achieved through communication in conscientiously constructed graphs. In large-scale linear parameter learning with Gaussian measurements, Gaussian Belief Propagation (Gaussian BP) ([2]) is an efficient distributed algorithm for calculating the marginal means of the unknown parameters.

In this work, Gaussian BP is used in both its synchronous and asynchronous variants on a security model. More specifically, in our system, there is a WiFi transmitter

in a room of an interior building and several wireless receivers/sensors that allow for a privacy-non intrusive security application of channel estimation. The distributed sensors exchange messages with one another using asynchronous GBP, in order to infer the wireless channel. The goal is to detect whether or not a moving person is in the room only by comparison of the variations of the channel between consecutive measurements of each sensor, without utilizing other means, such as microphones, cameras or infrared sensors. The aforementioned setting is depicted in Figure 1 and will be utilized in Section 5.1.2.

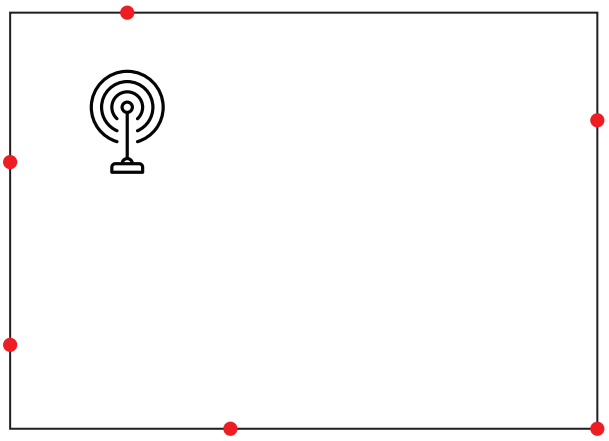


Figure 1.1: Room with WiFi antenna (transmitter) and sensors (receivers) located at various distances from the antenna. The sensors are represented with red dots.

Using appropriate metrics, we compare the results of the channel estimation using MMSE and LS estimators and both synchronous and asynchronous variants of the GBP, in cases where motion has occurred compared to the static case. By noticing a significant change in the channel estimate metrics, in cases where mobility occurs, a suitable threshold value can be selected; channel estimate changes that exceed the threshold value will indicate the presence of a moving person. On the other hand, channel estimate changes that are less than the threshold value will signify that no moving person is present in the room. Our proposed technique is significant since WiFi continues to be predominant in the Internet-Of-Things era and thus, no further equipment is needed.

Thesis Outline

In Chapter 2 we present the theoretical framework of inference algorithms and probabilistic graphical models, in Chapter 3 we study distributed inference using GBP and we present an MMSE application of GBP, in Chapter 4 we explain the proposed security model and we state two centralized estimators that will be used, in Chapter 5 we present the main numerical results and in Chapter 6 we state the main contributions of this work.

Notation

Boldface (lower case) is used for column vectors, $\mathbf{x}$, and (upper case) for matrices, $\mathbf{X}$. Let $\mathbf{X}^T$, $\mathbf{X}^H$, and $\mathbf{X}^*$ denote the transpose, the conjugate transpose, and the conjugate of $\mathbf{X}$, respectively. The Kronecker product of two matrices $\mathbf{X}$ and $\mathbf{Y}$ is denoted $\mathbf{X} \otimes \mathbf{Y}$, $\text{vec}(\mathbf{X})$ is the column vector obtained by stacking the columns of $\mathbf{X}$, $\text{tr}\{\mathbf{X}\}$ is the matrix trace, and $\text{diag}(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ is the N-by-N diagonal matrix with $\mathbf{x}_1, \ldots, \mathbf{x}_N$ at the main diagonal. The squared Frobenius norm of a matrix $\mathbf{X}$ is denoted $\|\mathbf{X}\|_F^2$ and is defined as the sum of the squared absolute values of all the elements. $\mathcal{CN}(\mathbf{0}, \mathbf{R})$ is used to denote circularly symmetric complex Gaussian random vectors, which has zero mean and $\mathbf{R}$ the covariance matrix. The notation $\triangleq$ is used for definitions. $\rho(\mathbf{A})$ denotes the spectral radius of matrix $\mathbf{A}$, namely $\rho(\mathbf{A}) = \max\{|\lambda_1|, |\lambda_2|, \ldots, |\lambda_m|\}$, where $\lambda_1, \ldots, \lambda_m$ are the eigenvalues of $\mathbf{A}$. $\mathbf{A} \succ \mathbf{0}$ denotes that $\mathbf{A}$ is positive definite.

Chapter 2

Inference Algorithms and Probabilistic Graphical Models

Inference is the task of extracting the probability of one or more random variables (hidden state) based on available observations.

Consider a collection of n random variables $\mathbf{x} = (x_1, \ldots, x_N)$ and observations $\mathbf{y} = (y_1, \ldots, y_N)$, where each of these random variables x_i, $1 \leq i \leq N$, take on a value in the set $\mathcal{X}$ and each observation variable y_i, $1 \leq i \leq N$, take on a value in the set $\mathcal{Y}$. Given observations $\mathbf{y}$, our goal is to answer questions about the distribution of $\mathbf{x}$ [3]. Given this setup, there are two primary computation tasks:

1. Calculating *posterior* beliefs,

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{y})} = \frac{p(\mathbf{x}, \mathbf{y})}{\sum_{\mathbf{x}' \in \mathcal{X}^N} p(\mathbf{x}', \mathbf{y})}. \tag{2.1}$$

   Generally, computation of the denominator of Equation 2.1 is expensive because, without any additional structure, a distribution over N variables, with each variable taking on values in $\mathcal{X}$, requires storing a table of size $|\mathcal{X}|^N$, where each entry contains

the probability of a particular realization. Therefore, calculation of *posterior* beliefs has complexity exponential in the number of variables N.

2. Calculating the *maximum a posteriori (MAP)* estimate,

$$\begin{aligned} \hat{\mathbf{x}} &\in \underset{\mathbf{x}\in\mathcal{X}^N}{\operatorname{argmax}}\, p(\mathbf{x}|\mathbf{y}) \\ &= \underset{\mathbf{x}\in\mathcal{X}^N}{\operatorname{argmax}}\, \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{y})} \\ &= \underset{\mathbf{x}\in\mathcal{X}^N}{\operatorname{argmax}}\, p(\mathbf{x},\mathbf{y}) \end{aligned} \tag{2.2}$$

As before, without any additional structure utilized, the aforementioned optimization problem requires searching over the entire space $\mathcal{X}^N$, for each possible value of $\mathbf{y}$ resulting in an exponential complexity in the number of variables N.

Considering a different scenario, where the N random variables are now independent, i.e.

$$p(x_1,\ldots,x_n) = p(x_1)\cdot\ldots\cdot p(x_N). \tag{2.3}$$

In this case, as posterior belief calculation can be done separately for each variable, the computational complexity of MAP estimation drops to $N\cdot|\mathcal{X}|$.

Therefore, we can exploit independence or some form of factorization, as it can compute efficiently both posterior beliefs and MAP estimation. By utilizing factorizations of joint probability distributions and representing them with graphical models, significant computational efficiency can be achieved.

Probabilistic Graphical Models (PGMs) provide a representation based on graphs and graph-like representations as the basis of encapsulating in an elegant way a complex probability distribution [3].

The three main types of PGMs are directed graphical models, undirected graphical models and factor graphs [4].

2.1 Directed Graphical Models

A directed graphical model (or Directed Acyclic Graph (DAG) or "Bayesian Network") is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ which consists of nodes $\mathcal{V}$, that represent random variables, and the directed edges between them denoted $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. The notation $(i, j) \in \mathcal{E}$ implies that there is a directed edge from i to j. We consider a *topological ordering* of the node (since $\mathcal{G}$ is directed and acyclic), according to which any node i comes after all of its parents. Then, the graph $\mathcal{G}$ implies the conditional independence

$$x_i \perp x_{\nu_i} | x_{\pi_i}, \tag{2.4}$$

where ν_i is the set of nodes that are not parents of i but they appear in the topological ordering before i. Hence, DAGs define families of distributions that factor by functions of nodes and their parents. In particular, we assign to each node i a random variable x_i and a non-negative-valued function $f_i(x_i, x_{\pi_i})$ such that,

$$\sum_{x_i \in \mathcal{X}} f_i(x_i, x_{\pi_i}) = 1,$$
$$\prod_i f_i(x_i, x_{\pi_i}) = p(x_1, \ldots, x_N),$$

where π_i denotes the set of parents of node i. Since the graph is acyclic, thus we must have $f_i(x_i, x_{\pi_i}) = p(x_i | x_{\pi_i})$.

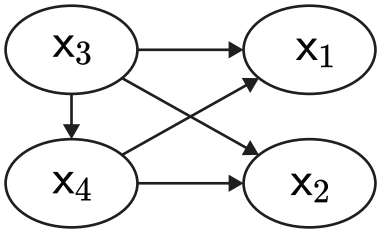


Figure 2.1: An example of a directed acyclic graph representing a distribution with 4 random variables.

2.2 Undirected Graphical Models

An undirected graphical model or "Markov Random Fields" (MRF) is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ are its nodes which represent random variables, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ are its undirected edges. These types of graphs define a family of probability distributions that satisfy the following graph separation property,

$$x_A \perp x_B | x_C, \tag{2.5}$$

whenever there is no path from any node in A to any node in B which does not pass through any node in C. A difference between directed and undirected graphical models is that undirected graphical models do not have a natural factorization into a product of conditional probabilities. Instead, we represent the distribution as a product of functions called *potentials*, times a normalization constant. Given a set of variables $x_1, \ldots, x_N$ and a set $\mathcal{C}$ of maximal cliques, we can define the following representation of the joint distribution:

$$p(\mathbf{x}) \propto \prod_{C \in \mathcal{C}} \psi(x_C) \tag{2.6}$$

$$= \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi(x_C), \tag{2.7}$$

where Z is called the *partition function* and is chosen such that the probabilities corresponding to all joint assignments sum to 1,

$$Z = \sum_{\mathbf{x}} \prod_{C \in \mathcal{C}} \psi(x_C). \tag{2.8}$$

2.3 Factor Graphs

A factor graph [5] consists of a vector of random variables $\mathbf{x} = (x_1, \ldots, x_n)$ and a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, which in addition to variable nodes also has factor nodes $\mathcal{F}$. The joint

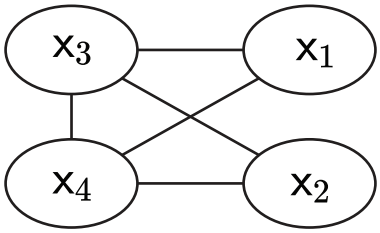


Figure 2.2: An example of an MRF representing a distribution with 4 random variables.

probability distribution associated to a factor graph is given by

$$p(x_1, \ldots, x_n) \propto \prod_{j \in J} f_j(X_j), \tag{2.9}$$

where J is a discrete index set, X_j is a subset of $\{x_1, \ldots, x_n\}$ and $f_j(X_j)$ is a function having X_j as input.

The factor graph is a bipartite graph between variable nodes and factor node, i.e. there are no edges connecting a factor node with another factor node or a variable node with another variable node, that expresses the structure of the factorization 2.9. A factor graph has a variable node for each variable x_i, a factor node for each local function f_i and an edge that connects a variable node to a factor node if and only if x_i is an argument of f_i. Finally, the constraint that is imposed on the factors is that they must be non-negative.

In order to better comprehend what factor graphs are, we provide an example. Let p be a probability density function that can be expressed as the product

$$p(x_1, x_2, x_3, x_4) \propto f_1(x_1, x_3) \cdot f_2(x_1, x_3, x_4) \cdot f_3(x_2, x_3, x_4) \tag{2.10}$$

of two factors, so that $J = \{1, 2, 3\}$, $X_1 = \{x_1, x_3\}$, $X_2 = \{x_1, x_3, x_4\}$ and $X_3 = \{x_2, x_3, x_4\}$. The factor graph that corresponds to p is shown in Figure 2.3.

Factor graphs are seminal in this work, since Gaussian Belief Propagation is implemented as iterative message passing between the factor graph's nodes.

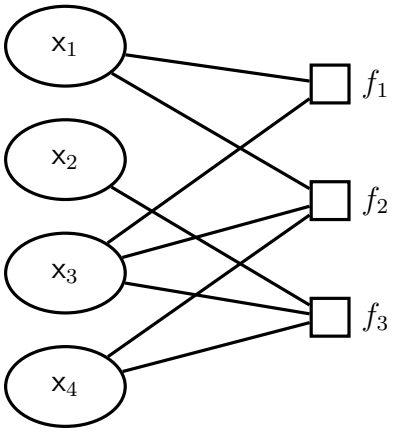


Figure 2.3: A factor graph for the product $p(x_1, x_2, x_3, x_4) \propto f_1(x_1, x_3) \cdot f_2(x_1, x_3, x_4) \cdot f_3(x_2, x_3, x_4)$.

The variable nodes are represented with circles and the factor node with squares.

Chapter 3

Distributed Inference with Gaussian Belief Propagation (GBP)

3.1 Gaussian Belief Propagation under High-Order Factorization and Asynchronous Scheduling

Let a Gaussian random vector $\mathbf{x} = [x_1, x_2, \ldots, x_n]^\top \in \mathbb{R}^n$ with the following probability density function (pdf):

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{N/2}|\Lambda|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \Lambda^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\} \tag{3.1}$$

$$\propto \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \Lambda^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}, \tag{3.2}$$

where $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Lambda})$, with mean $\boldsymbol{\mu} = \mathbb{E}[\mathbf{x}]$ and covariance matrix $\boldsymbol{\Lambda} = \mathbb{E}[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^\top]$. This form of representation is called the *covariance form*.

The *information form* is yet another extremely helpful format, where the pdf is expressed as:

$$p(\mathbf{x}) \propto \exp\left\{-\frac{1}{2}\mathbf{x}^\top \mathbf{J}\mathbf{x} + \mathbf{h}^\top \mathbf{x}\right\}, \tag{3.3}$$

where $\mathbf{x} \sim \mathcal{N}^{-1}(\mathbf{h}, \mathbf{J})$, with potential $\mathbf{h}$ and information (or precision) matrix $\mathbf{J}$. In this form $\mathbf{J} = \mathbf{\Lambda}^{-1} \succ \mathbf{0}$ and $\mathbf{h} = \mathbf{J}\boldsymbol{\mu}$.

The *information form* is often preferred over the *covariance form* as the posterior can be easily formed from the factors. Moreover, the information matrix is sparse and closely resembles the factor graph's structure.

Conventionally, Gaussian BP is performed under a pairwise factorization of the joint Gaussian pdf:

$$p(\mathbf{x}) \propto \prod_{i} \Phi_i\left(x_i\right) \prod_{i \neq j} \Phi_{ij}\left(x_i, x_j\right), \tag{3.4}$$

where $\Phi_i\left(x_i\right)$ is a function that only depends on x_i while $\Phi_{ij}\left(x_i, x_j\right)$ is a function that depends on x_i and x_j.

Following the work in [6], the joint Gaussian pdf can be expressed as

$$p(\mathbf{x}) \propto \prod_{i=1}^{n} f_i\left(x_i\right) \prod_{j=1}^{m} g_j\left(\mathcal{X}_j\right), \tag{3.5}$$

where $f_i(x_i)$ is a function of x_i and $g_j(\mathcal{X}_j)$ is a function of a set of variables $\mathcal{X}_j \subseteq \{x_1, x_2, \ldots, x_n\}$. We refer to the *high-order factorization* of the joint Gaussian pdf in the case where at least one function $g_j(\mathcal{X}_j)$ contains more than two variables. We consider a high-order factorization with $\mathbf{J} = \mathbf{\Lambda} + \mathbf{\Xi}^\top \mathbf{\Sigma} \mathbf{\Xi}$ and $\mathbf{h} = \mathbf{\Lambda}\boldsymbol{\xi} + \mathbf{\Xi}^\top \mathbf{\Sigma}\mathbf{u}$, where $\mathbf{\Lambda} \triangleq diag(\eta_1, \eta_2, \ldots, \eta_n)$, $\mathbf{\Sigma} \triangleq diag(\zeta_1, \zeta_2, \ldots, \zeta_m)$ with $\eta_i \geq 0$, $\zeta_j > 0$, $\mathbf{\Xi} \in \mathbb{R}^{m \times n}$, $\boldsymbol{\xi} \in \mathbb{R}^n$ and $\mathbf{u} \in \mathbb{R}^m$. Under this factorization, the joint Gaussian pdf can be rewritten as

$$\begin{aligned}
p(\mathbf{x}) &\propto \exp\left\{-\frac{1}{2}\mathbf{x}^\top \left(\mathbf{\Lambda} + \mathbf{\Xi}^\top \mathbf{\Sigma} \mathbf{\Xi}\right) \mathbf{x} + \left(\mathbf{\Lambda}\boldsymbol{\xi} + \mathbf{\Xi}^\top \mathbf{\Sigma}\mathbf{u}\right)^\top \mathbf{x}\right\} \\
&\propto \exp\left\{-\frac{1}{2}\left(\mathbf{x} - \boldsymbol{\xi}\right)^\top \mathbf{\Lambda}\left(\mathbf{x} - \boldsymbol{\xi}\right)\right\} \exp\left\{-\frac{1}{2}\left(\mathbf{\Xi}\mathbf{x} - \mathbf{u}\right)^\top \mathbf{\Sigma}\left(\mathbf{\Xi}\mathbf{x} - \mathbf{u}\right)\right\} \\
&\propto \prod_{i=1}^{n} \exp\left\{-\frac{1}{2}\eta_i \left(x_i - \xi_i\right)^2\right\} \prod_{j=1}^{m} \exp\left\{-\frac{1}{2}\zeta_i \left(\mathbf{\Xi}_{j,:}\mathbf{x} - u_j\right)^2\right\},
\end{aligned} \tag{3.6}$$

where $\mathbf{\Xi}_{j,:}$ denotes the j-th row of $\mathbf{\Xi}$. Based on Equations 3.5 and 3.6, we have

$$f_i(x_i) \propto \exp\left\{-\frac{1}{2}\eta_i(x_i-\xi_i)^2\right\} \tag{3.7}$$

$$g_j(\mathcal{X}_j) \propto \exp\left\{-\frac{1}{2}\zeta_i\left(\sum_{k\in\mathcal{V}_j}\Xi_{jk}x_k - u_j\right)^2\right\} \tag{3.8}$$

where $\mathcal{X}_j \triangleq \{x_k|\Xi_{jk}\neq 0\}$ and $\mathcal{V}_j \triangleq \{k|x_k\in\mathcal{X}_j\}$. More specifically, $\mathcal{X}_j$ denotes the set of variables that are connected to factor j and $\mathcal{V}_j$ denotes the set of indices of those variables. We also define $\mathcal{G}_i$, which is the set of indices of factors $\{g_j\}_{j=1}^m$ connected directly to variable x_i in the factor graph.

Since we examine GBP on factor graphs, which are bipartite, i.e. we have only factor-to-variable node connections and variable-to-factor node connections, the expressions of GBP under high-order factorization only require factor-to-variable messages, $m^{(l)}_{g_j\to x_i}(x_i)$, and variable-to-factor messages, $m^{(l)}_{x_i\to g_j}(x_i)$, at each iteration (l). The message update rules of the algorithm under synchronous scheduling are

$$m^{(l)}_{g_j\to x_i}(x_i) \propto \int_{-\infty}^{+\infty} g_j(\mathcal{X}_j)\prod_{k\in\mathcal{V}_j\setminus i} m^{(l-1)}_{x_k\to g_j}(x_k)\, d\mathcal{X}_j\setminus x_i \tag{3.9}$$

$$m^{(l)}_{x_i\to g_j}(x_i) \propto f_i(x_i)\prod_{k\in\mathcal{G}_i\setminus j} m^{(l)}_{g_k\to x_i}(x_i). \tag{3.10}$$

By inserting the expression of $m^{(l)}_{x_i\to g_j}(x_i)$ in 3.10 into 3.9, we obtain

$$m^{(l)}_{g_j\to x_i}(x_i) \propto \int_{-\infty}^{+\infty} g_j(\mathcal{X}_j)\prod_{k\in\mathcal{V}_j\setminus i}\left(f_k(x_k)\prod_{k'\in\mathcal{G}_k\setminus j} m^{(l)}_{g_{k'}\to x_i}(x_k)\right) d\mathcal{X}_j\setminus x_i. \tag{3.11}$$

TWithout loss of generality, we assume that the factor-to-variable messages $m^{(l-1)}_{g_{k'}\to x_i}(x_k)$ are of Gaussian form with the expression $m^{(l-1)}_{g_{k'}\to x_i}(x_k) \sim \mathcal{N}\left(x_k; \mu^{(l-1)}_{g_{k'}\to x_i}, \frac{1}{\nu^{(l-1)}_{g_{k'}\to x_i}}\right)$, where $\mu^{(l-1)}_{g_{k'}\to x_i}$ and $\nu^{(l-1)}_{g_{k'}\to x_i}$ their mean and precision, respectively. Therefore, substituting the expressions of $f_i(x_i)$ and $g_j(\mathcal{X}_j)$ in Eq. 3.7 and 3.8 respectively, and the Gaussian form of $m^{(l-1)}_{g_{k'}\to x_i}(x_k)$ in Eq. 3.11 we get the analytical expression of factor-to-variable node

messages as:

$$m^{(l)}_{g_j\to x_i}(x_i) \propto \int_{-\infty}^{+\infty} \exp\left\{-\frac{1}{2}\zeta_j\left(\sum_{k\in\mathcal{V}_j}\Xi_{jk}x_k - u_j\right)^2\right\}$$
$$\prod_{k\in\mathcal{V}_j\backslash i}\exp\left\{-\frac{1}{2}\left(\eta_k + \sum_{k'\in G_k\backslash k}\nu^{(l-1)}_{g_{k'}\to x_k}\right)x_k^2 + \left(\eta_k\xi_k + \sum_{k'\in\mathcal{G}_k\backslash j}\nu^{(l-1)}_{g_{k'}\to x_k}\mu^{(l-1)}_{g_{k'}\to x_k}\right)x_k\right\} d\mathcal{X}_j\backslash x_i, \tag{3.12}$$

which, in any case, maintain their Gaussian form [6], with $m^{(l)}_{g_j\to x_i}(x_i) \sim \mathcal{N}\left(x_i; \mu^{(l)}_{g_j\to x_i}, 1/\nu^{(l)}_{g_j\to x_i}\right)$ with the following mean and precision:

$$\mu^{(l)}_{g_j\to x_i} = \begin{cases} \Xi^{-1}_{ji}u_j - \sum_{k\in\mathcal{V}_j\backslash i}\dfrac{\Xi^{-1}_{ji}\Xi_{jk}\left(\eta_k\xi_k + \sum_{k'\in\mathcal{G}_k\backslash j}\nu^{(l-1)}_{g_{k'}\to x_k}\mu^{(l-1)}_{g_{k'}\to x_k}\right)}{\eta_k + \sum_{k'\in\mathcal{G}_k\backslash j}\nu^{(l-1)}_{g_{k'}\to x_k}}, \\ \qquad\qquad \text{if } \eta_k + \sum_{k'\in G_k\backslash k}\nu^{(l-1)}_{g_{k'}\to x_k} > 0, \forall k\in\mathcal{V}_k\backslash i \\ 0, \qquad\qquad \text{otherwise} \end{cases} \tag{3.13}$$

$$\nu^{(l)}_{g_j\to x_i} = \frac{\Xi^2_{ji}}{\zeta_j^{-1} + \sum_{k\in\mathcal{V}_j\backslash i}\Xi^2_{jk}\left(\eta_k + \sum_{k'\in\mathcal{G}_k\backslash j}\nu^{(l-1)}_{g_{k'}\to x_k}\right)^{-1}}, \tag{3.14}$$

respectively.

The aforementioned equations demonstrate that in order to completely describe each message $m^{(l)}_{g_j\to x_i}(x_i)$ at iteration (l) only its *mean* $\mu^{(l)}_{g_j\to x_i}$ and its *precision* $\nu^{(l)}_{g_j\to x_i}$ are necessary. Additionally, it is evident that the calculation of each $\nu^{(l)}_{g_j\to x_i}$ only requires other precision parameters $\nu^{(l)}_{g_k\to x_p}$. On the other hand, in order to calculate $\mu^{(l)}_{g_j\to x_i}$, both different precisions $\nu^{(l)}_{g_k\to x_p}$ and means $\mu^{(l)}_{g_k\to x_p}$ are required.

Using the updated message $m^{(l)}_{g_j\to x_i}$ and $f_i(x_i)$, the BP belief, $b^{(l)}(x_i)$, of x_i at iteration (l) can be computed by

$$b^{(l)}(x_i) \propto f_i(x_i)\prod_{k\in\mathcal{G}_i} m^{(l)}_{g_k\to x_i}(x_i). \tag{3.15}$$

By inserting the expressions of $f_i(x_i)$ (Equation 3.7) and the Gaussian form of $m^{(l)}_{g_k \to x_i}(x_i)$ into Equation 3.15, we obtain

$$b^{(l)}(x_i) \propto \exp\left\{ -\frac{1}{2}\left(\eta_i + \sum_{k \in \mathcal{G}_i} v^{(l)}_{g_k \to x_i} \right) x_i^2 + \left(\eta_i \xi_i + \sum_{k \in \mathcal{G}_i} v^{(l)}_{g_k \to x_i} \mu^{(l)}_{g_k \to x_i} \right) x_i \right\}. \tag{3.16}$$

According to [6], the BP beliefs are valid Gaussian pdfs, with $b^{(l)}(x_i) \sim \mathcal{N}\left(x_i; \epsilon_i^{(l)}, \sigma_i^{(l)}\right)$, where

$$\epsilon_i^{(l)} = \frac{\eta_i \xi_i + \sum_{k \in \mathcal{G}_i} \nu^{(l)}_{g_k \to x_i} \mu^{(l)}_{g_k \to x_i}}{\eta_i + \sum_{k \in \mathcal{G}_i} \nu^{(l)}_{g_k \to x_i}}, \tag{3.17}$$

$$\sigma_i^{(l)} = \frac{1}{\eta_i + \sum_{k \in \mathcal{G}_i} \nu^{(l)}_{g_k \to x_i}}. \tag{3.18}$$

On the assumption that $\nu^{(l)}_{g_j \to x_i}$ are initialized such that convergence is guaranteed, we can replace $\nu^{(l-1)}_{g_{k'} \to x_k}$ with $\nu^{*}_{g_{k'} \to x_k}$ in Equation 3.13. Then if $\boldsymbol{\mu}^{(l)}$ stacks all $\mu^{(l)}_{g_j \to x_i}$ at iteration (l), GBP can be reduced to the *synchronous* affine fixed point problem:

$$\boldsymbol{\mu}^{(l)} = \mathbf{A}\boldsymbol{\mu}^{(l-1)} + \mathbf{c}, \tag{3.19}$$

where $\mathbf{A}$ is an $|\mathcal{E}| \times |\mathcal{E}|$ matrix such that $\mathbf{A}\boldsymbol{\mu}^{(l-1)}$ is a column vector containing elements

$$\alpha_{ij} = \begin{cases} -\sum_{k \in \mathcal{V}_j \backslash i} \frac{\Xi_{ji}^{-1} \Xi_{jk} \sum_{k' \in \mathcal{G}_k \backslash j} \nu^{*}_{g_{k'} \to x_k} \mu^{(l-1)}_{g_{k'} \to x_k}}{\eta_k + \sum_{k' \in \mathcal{G}_k \backslash j} \nu^{*}_{g_{k'} \to x_k}} & , \text{ if } \eta_k + \sum_{k' \in \mathcal{G}_k \backslash j} \nu^{*}_{g_{k'} \to x_k} > 0, \forall k \in \mathcal{V}_j \backslash i \\ 0 & , \text{ otherwise} \end{cases} \tag{3.20}$$

and $\mathbf{c}$ is a column vector containing elements

$$\beta_{ij} = \begin{cases} -\sum_{k \in \mathcal{V}_j \backslash i} \frac{\Xi_{ji}^{-1} \Xi_{jk} \eta_k \xi_k}{\eta_k + \sum_{k' \in \mathcal{G}_k \backslash j} \nu^{*}_{g_{k'} \to x_k}} & , \text{ if } \eta_k + \sum_{k' \in \mathcal{G}_k \backslash j} \nu^{*}_{g_{k'} \to x_k} > 0, \forall k \in \mathcal{V}_j \backslash i \\ 0 & , \text{ otherwise} \end{cases} \tag{3.21}$$

The expressions of GBP under high-order factorization can be updated utilizing both *synchronous* and *asynchronous* scheduling. In fact, asynchronous updates might be able to converge even in cases where synchronous ones do not, as explained in Section 3.2.

3.2 Affine Fixed Point (AFP) Problem

According to eq. 3.19, it was shown that GBP can be reduced to the synchronous affine fixed point problem $\boldsymbol{\mu}^{(l)} = \mathbf{A}\boldsymbol{\mu}^{(l-1)} + \mathbf{c}$. In this section, we will study the convergence of the affine fixed point problem for both *synchronous* and *asynchronous* cases.

Consider the real vectors $\mathbf{x}^{(0)},\ \mathbf{b} \in \mathbb{R}^n$ and the real square matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$. The following recursion holds:

$$\mathbf{x}^{(l)} = \mathbf{A}\mathbf{x}^{(l-1)} + \mathbf{b}, \quad l = 1, 2, \dots \tag{3.22}$$

The solution of this problem is named the *fixed point* and it is denoted by $\mathbf{x}^* \triangleq \lim_{l\to\infty} \mathbf{x}^{(l-1)} = \lim_{l\to\infty} \mathbf{x}^{(l)}$.

Each element of the Equation 3.22 can be written as

$$x_k^{(l)} = \sum_{j=1}^{n} a_{kj} x_j^{(l-1)} + b_k, \quad l = 1, 2, \dots \quad k = 1, 2, \dots, n, \tag{3.23}$$

where $x_k^{(l)}$ and b_k denote the k-th element of $\mathbf{x}^{(l)}$ and $\mathbf{b}$, respectively, and a_{ij} the element of the i-th row and j-th column of $\mathbf{A}$. In order to calculate $x_k^{(l)}$, all variables $x_j^{(l-1)}$, with respective $a_{kj} \neq 0$, of the previous iteration $(l-1)$ must be known. This constraint leads to *synchronous scheduling*, since no element of the vector $\mathbf{x}$ can have its output updated during a specific iteration before all necessary input is readily available for all variables.

This strong requirement can be loosened. We can specifically assume that at iteration (l) not all elements of $\mathbf{x}^{(l-1)}$ are available in order to compute the corresponding elements of the update vector $\mathbf{x}^{(l)}$. Then, without waiting for all elements to be updated, we update the vector $\mathbf{x}^{(l)}$ while keeping the corresponding values from the previous iteration. As a result, we get what we will refer to as *asynchronous scheduling*.

In order to formally formulate the above, we adopt the notation of seminal work in

[7, 8, 6]. More specifically, at each iteration (l), we introduce the functions $\psi_k^{(l)}$, $\forall k \in \{1, 2, \ldots, n\}$, which are defined as

$$\psi_k^{(l)} = \begin{cases} 1, \text{ if } x_k \text{ is updated at iteration } (l), \\ 0, \text{ otherwise.} \end{cases} \tag{3.24}$$

Let $\boldsymbol{\psi}^{(l)}$ the binary vector created if we stack all $\left\{\psi_k^{(l)}\right\}$, $k \in \{1, 2, \ldots, n\}$, at iteration (l), and $\boldsymbol{\Psi}^{(l)} \triangleq \text{diag}\{\boldsymbol{\psi}^{(l)}\}$ the corresponding diagonal matrix with $\boldsymbol{\psi}^{(l)}$ at its diagonal. As a result, the asynchronous framework modifies the Equation 3.22 to

$$\begin{aligned} \mathbf{x}^{(l)} &= \boldsymbol{\Psi}^{(l)} \left(\mathbf{A}\mathbf{x}^{(l-1)} + \mathbf{b}\right) + \left(\mathbf{I} - \boldsymbol{\Psi}^{(l)}\right) \mathbf{x}^{(l-1)} \\ &= \left(\boldsymbol{\Psi}^{(l)}\mathbf{A} + \mathbf{I} - \boldsymbol{\Psi}^{(l)}\right) \mathbf{x}^{(l-1)} + \boldsymbol{\Psi}^{(l)}\mathbf{b}. \end{aligned} \tag{3.25}$$

Notice that if $\boldsymbol{\Psi}^{(l)} = \mathbf{I}$, $\forall l$, i.e. $\psi_k^{(l)} = 1$, $\forall l$ and $\forall k$, the asynchronous update of Equation 3.25 reduces to the synchronous one of Equation 3.22.

Based on [7, 8], we assume that the aforementioned functions $\psi_k^{(l)}$ are Bernoulli random variables, independent across the different iterations (l) but possibly dependent and not identically distributed across k, with parameters p_k, $\forall k \in \{1, 2, \ldots, n\}$. As a result, we can define the expected value of matrices $\boldsymbol{\Psi}^{(l)}$, $\mathbb{E}[\boldsymbol{\Psi}^{(l)}] \triangleq \mathbf{P} = \text{diag}\{\mathbf{p}\}$, where $\mathbf{p} = \mathbb{E}[\boldsymbol{\psi}^{(l)}]$, a quantity that as we will see plays a major role in the convergence of recursion in Equation 3.25.

3.2.1 Convergence Conditions

3.2.1.1 Convergence of Synchronous AFP Problem

According to [9, 10], the convergence of the *synchronous* affine fixed point problem of Equation 3.22 is entirely determined by the spectral radius of the matrix $\mathbf{A}$. More

specifically, it is shown that a *necessary* and *sufficient* condition in order for Equation 3.22 to converge is

$$\rho(\mathbf{A}) < 1. \tag{3.26}$$

3.2.1.2 Convergence of Asynchronous AFP Problem

The topic of the convergence properties of the *asynchronous* affine update of Equation 3.25 is a trickier one. Seminal works in [11, 12, 13], which adopt a state-space recursions' perspective to approach the AFP problem, while [6] studies the convergence of an asynchronous variant of Gaussian Belief Propagation. These works offer *sufficient* conditions for mean convergence which are directly applicable to our model.

A *necessary* and *sufficient* condition for convergence in the mean sense, i.e., convergence of $\lim_{l\to\infty} \mathsf{E}\left[\mathbf{x}^{(l)}\right]$ is

$$\rho(\bar{\mathbf{A}}) < 1, \tag{3.27}$$

where $\bar{\mathbf{A}} = \mathbf{P}(\mathbf{A}-\mathbf{I})+\mathbf{I}$ [6, 11, 12]. In addition, given that $\rho(\bar{\mathbf{A}}) < 1$, a necessary and sufficient condition for convergence in the mean square sense, i.e., covariance matrix of the error vector approaches the all-zero matrix for $l \to \infty$ is

$$\rho(\mathbf{S}) < 1, \tag{3.28}$$

where $\mathbf{S} = \bar{\mathbf{A}}\otimes\bar{\mathbf{A}}+((\mathbf{I}-\mathbf{P}))\otimes\mathbf{P})\,\mathbf{J}\,((\mathbf{A}-\mathbf{I})\otimes(\mathbf{A}-\mathbf{I}))$ [11, 12], with $\mathbf{J} = \sum_{i=1}^{n}\left(\mathbf{e}_i\mathbf{e}_i^{\mathrm{H}}\right)\otimes\left(\mathbf{e}_i\mathbf{e}_i^{\mathrm{H}}\right) = \mathrm{diag}(\mathrm{vec}(\mathbf{I})) \in \mathbb{R}^{n^2\times n^2}$, $\mathbf{e}_i \in \mathbb{R}^n$ the i-th standard vector that has 1 at the i-th index and 0 elsewhere and $\otimes$ the Kronecker product.

A convergent synchronous AFP problem may diverge when implemented asynchronously, and conversely, a divergent asynchronous AFP problem may converge simply by the use of asynchronicity. The main difference lies in the notion of convergence. The asynchronous

case considers the average behavior and focuses on a mean sense of convergence. The notion of convergence is more relaxed in the asynchronous case and therefore the condition is more relaxed as well. [11]

It is important to note that update probabilities play an important role in the stability of randomized updates. Namely, the randomized system can be stable for some set of probabilities, and it may get unstable for some other set of probabilities. In fact, the rate of convergence can also be increased with the optimal selection of the probabilities similar to other randomized algorithms. [11]

3.3 MMSE Application of GBP

Let $\mathbf{x} \in \mathbb{R}^n$ a real Gaussian vector such that $\mathbf{x} \sim \mathcal{N}(\mathbf{x}; \mathbf{0}, \mathbf{W})$, where $\mathbf{W} \succ \mathbf{O}$. Consider the system model

$$\mathbf{y} = \mathbf{B}\mathbf{x} + \mathbf{z}, \tag{3.29}$$

where $\mathbf{B} \in \mathbb{R}^{m \times n}$ a real matrix and $\mathbf{z} \in \mathbb{R}^m$ a Gaussian vector independent of $\mathbf{x}$, with $\mathbf{z} \sim \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{R})$ and $\mathbf{R} \succ \mathbf{O}$. Then, assuming that we only have access to the noisy measurements Eq. 3.29, the Linear Minimum Mean Square Error (LMMSE) estimator of $\mathbf{x}$ is given by [1], [14], [15]

$$\hat{\mathbf{x}} = \left(\mathbf{W}^{-1} + \mathbf{B}^\top \mathbf{R}^{-1} \mathbf{B}\right)^{-1} \mathbf{B}^\top \mathbf{R}^{-1} \mathbf{y}. \tag{3.30}$$

Similarly to the case of the linear system of equations of the previous Section, if we set $\mathbf{\Lambda} = \mathbf{W}^{-1}$, $\mathbf{\Xi} = \mathbf{B}$, $\mathbf{\Sigma} = \mathbf{R}^{-1}$, $\boldsymbol{\xi} = \mathbf{0}$ and $\mathbf{u} = \mathbf{y}$ in Eq. 3.6 we get the distribution

$$
\begin{aligned}
p(\mathbf{x}) &= \mathcal{N}\left(\left(\mathbf{W}^{-1}+\mathbf{B}^{\top}\mathbf{R}^{-1}\mathbf{B}\right)^{-1}\mathbf{B}^{\top}\mathbf{R}^{-1}\mathbf{y}, \mathbf{W}^{-1}+\mathbf{B}^{\top}\mathbf{R}^{-1}\mathbf{B}\right) \\
&\propto \exp\left\{-\frac{1}{2}\mathbf{x}^{\top}\left(\mathbf{W}^{-1}+\mathbf{B}^{\top}\mathbf{R}^{-1}\mathbf{B}\right)\mathbf{x}+\mathbf{y}^{\top}\left(\mathbf{R}^{-1}\right)^{\top}\mathbf{B}\mathbf{x}\right\}.
\end{aligned}
\tag{3.31}
$$

Utilizing Gaussian Belief Propagation, we can infer the expected value of $p(\mathbf{x})$, which will ultimately yield the desired estimate $\hat{\mathbf{x}}$.

Chapter 4

Inference and Security

4.1 Security Idea

Consider a flat block-fading MIMO system with a transmitter equipped with M transmit antennas at the Access Point (AP) and N receive single-antenna terminals. When there is no obstacle between the transmitter and the receiver, the channel matrix is expected to remain relatively constant. However, in the case where a human is between the transmitter and the receiver, the power of the signal will be attenuated and we expect the channel matrix to be modified. Our goal is to estimate the channel matrix at two consecutive time instances t_1 and t_2 and infer whether or not human mobility occurred in this time period in a non-privacy intrusive way just by comparing the channel state information between those two moments. In the next section, we will explain in detail the algorithm that was utilized for this purpose. Detection of human presence using radio frequency signals, even without line-of-sight (LOS) measurements has been explored in the radar community and examples can be found in [16] and references therein.

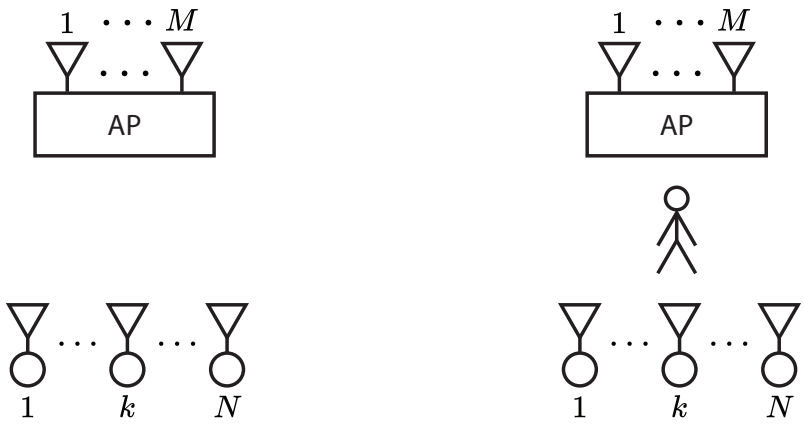


Figure 4.1: System model instance without (left) and with (right) human presence

4.2 Inference Algorithm

The $N \times 1$ complex received signal vector can be modeled as [17]

$$\bar{\mathbf{y}}_i = \mathbf{H}\bar{\mathbf{p}}_i + \bar{\mathbf{v}}_i, \tag{4.1}$$

where $\mathbf{H}$ is the $N \times M$ complex random channel matrix, $\bar{\mathbf{p}}_i$ is the $M \times 1$ complex vector of the transmitted signals, and $\bar{\mathbf{v}}_i$ is the $N \times 1$ complex zero-mean white noise vector.

It should be emphasized that in any statistical expectation below, the matrix $\mathbf{H}$ is considered random. At the same time, any estimator of $\mathbf{H}$ should also be able to estimate a specific realization of this random matrix that corresponds to the present block of received data.

For M transmit antennas, in order to estimate the channel matrix $\mathbf{H}$, we consider $M_{T_x} \geq M$ training signal vectors $\bar{\mathbf{p}}_1, \ldots, \bar{\mathbf{p}}_{M_{T_x}}$ be transmitted. The corresponding $N \times M_{T_x}$ matrix $\mathbf{Y} \triangleq [\bar{\mathbf{y}}_1, \ldots, \bar{\mathbf{y}}_{M_{T_x}}]$ of the received signals can be expressed as [17, 18]

$$\mathbf{Y} = \mathbf{H}\mathbf{P} + \mathbf{V}, \tag{4.2}$$

where

$$\mathbf{P} = [\bar{\mathbf{p}}_1, \ldots, \bar{\mathbf{p}}_{M_{T_x}}] \tag{4.3}$$

is the $M \times M_{T_x}$ training matrix and $\mathbf{V} = [\bar{\mathbf{v}}_1, \cdots, \bar{\mathbf{v}}_{M_{T_x}}]$ is the $N \times M_{Tx}$ sensor noise matrix that is uncorrelated to the channel $\mathbf{H}$. The combined noise matrix is modeled as a $\mathrm{vec}(\mathbf{V}) \in \mathcal{CN}(\mathbf{0}, \mathbf{S})$, where $\mathbf{S} \in \mathbb{C}^{N \cdot M_{T_x} \times N \cdot M_{T_x}}$ is the positive definite covariance matrix.

Since the multipath propagation is treated as quasi-static block fading, the channel realization $\mathbf{H}$ is independent of earlier channel estimations and remains constant during the training transmission.

The deterministic large-scale channel path-loss model listed below is used [19]:

$$\mathsf{L}_{\mathrm{k}} = \left(\frac{\lambda}{4\pi d_0}\right)^2 \left(\frac{d_0}{d_{\mathrm{k}}}\right)^{v_{\mathrm{k}}}, \tag{4.4}$$

where k signifies the $\mathrm{k}-th$ receiver, λ is the carrier wavelength, d_0 is a reference distance and v_{k} is the path-loss exponent for link k.

Each element of the channel $\mathbf{H}$ of dimension $N \times M$ is a complex number, i.e. $H_{i,j} = x + jy = h_{ij}$, which follows the Rayleigh distribution where x and y are i.i.d. with $x, y \sim \mathcal{N}(0, \frac{\sigma^2}{2})$. The mean value of $|h_{ij}|^2$ is:

$$\mathbb{E}[|h_{ij}|^2] = \mathbb{E}[x^2 + y^2] = \frac{\sigma^2}{2} + \frac{\sigma^2}{2} = \sigma^2 \tag{4.5}$$

The matrix $\mathbf{H}$ consists of the following elements: $H_{ij} = \sqrt{L_{ij}} h_{ij}$.

The noise of the system is denoted: $\mathbf{V} = [\bar{\mathbf{v}}_1, \cdots, \bar{\mathbf{v}}_{M_{T_x}}]$. Each vector $\bar{\mathbf{v}}_k$ has N v_i elements where:

$$v_i \sim \mathcal{CN}(\mathbf{0}, \sigma_n^2 \boldsymbol{I}) \tag{4.6}$$

$$\sim \mathcal{CN}(\mathbf{0}, N_0 W T_s \boldsymbol{I}) \tag{4.7}$$

$$\sim \mathcal{CN}(\mathbf{0}, L N_0 \boldsymbol{I}) \tag{4.8}$$

$$\sim \mathcal{CN}(\mathbf{0}, LkT_0\boldsymbol{I}), \tag{4.9}$$

where L is a properly chosen oversampling factor, k is the Boltzmann constant and T_0 is the room temperature in Kelvin.

The components in the ideal training matrix should all have the same magnitude if the peak transmitted power per antenna is constrained and equal to $\mathcal{P}$. A properly normalized submatrix of the discrete Fourier transform (DFT) matrix can be employed to fulfill this criterion, as defined below [17]:

$$\mathbf{P} = \sqrt{\frac{\mathcal{P}}{M \times M_{Tx}}} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & W_{M_{Tx}} & \cdots & W_{M_{Tx}}^{M_{Tx}-1} \\ \vdots & \vdots & & \vdots \\ 1 & W_{M_{Tx}}^{M-1} & \cdots & W_{M_{Tx}}^{(M-1)(M_{Tx}-1)} \end{bmatrix}, \tag{4.10}$$

where $W_{M_{Tx}} = e^{j2\pi/M_{Tx}}$.

A channel estimation algorithm's task is to recover the channel matrix $\mathbf{H}$ based on the knowledge of $\mathbf{S}$ and $\mathbf{P}$. For this purpose, we use the Least Squares (LS) and the Minimum Mean Squared Error (MMSE) estimators. In order to use the aforementioned estimators, we must perform vectorization of $\mathbf{H}$, since their inputs must be vectors whereas we want to estimate a matrix.

4.2.1 LS Estimator

We use the LS method to estimate the realization of the channel matrix given $\mathbf{P}$ and the received data, according to which [17]:

$$\hat{\mathbf{H}}_{\text{LS}} = \mathbf{Y}\mathbf{P}^{\dagger}, \tag{4.11}$$

where $\mathbf{P}^{\dagger} = \mathbf{P}^{H}(\mathbf{P}\mathbf{P}^{H})^{-1}$ is the pseudoinverse of $\mathbf{P}$.

4.2.2 MMSE Estimator

According to [18], by vectorizing the received signal in equation 4.2 and applying $\text{vec}(\mathbf{ABC}) = (\mathbf{C}^T \otimes \mathbf{A})\text{vec}(\mathbf{B})$, the received training signal of our system can be expressed as

$$\text{vec}(\mathbf{Y}) = \widetilde{\mathbf{P}}\text{vec}(\mathbf{H}) + \text{vec}(\mathbf{V}), \tag{4.12}$$

where $\widetilde{\mathbf{P}} \triangleq \mathbf{P}^T \otimes \mathbf{I}$.

The MMSE estimator, $\widehat{\mathbf{H}}_{\text{MMSE}}$, of the Rayleigh fading channel matrix $\mathbf{H}$ is [18]:

$$\text{vec}(\widehat{\mathbf{H}}_{MMSE}) = (\mathbf{R}^{-1} + \widetilde{\mathbf{P}}^H\mathbf{S}^{-1}\widetilde{\mathbf{P}})^{-1}\widetilde{\mathbf{P}}^H\mathbf{S}^{-1}\text{vec}(\mathbf{Y}) \tag{4.13}$$

$$= \mathbf{R}\widetilde{\mathbf{P}}^H(\widetilde{\mathbf{P}}\mathbf{R}\widetilde{\mathbf{P}}^H + \mathbf{S})^{-1}\text{vec}(\mathbf{Y}). \tag{4.14}$$

The error covariance $\mathbf{C}_{\text{MMSE}} \triangleq \mathbb{E}\{(\text{vec}(\mathbf{H}) - \text{vec}(\widehat{\mathbf{H}}_{\text{MMSE}}))(\text{vec}(\mathbf{H}) - \text{vec}(\widehat{\mathbf{H}}_{\text{MMSE}}))^H\}$ becomes

$$\mathbf{C}_{\text{MMSE}} = (\mathbf{R}^{-1} + \widetilde{\mathbf{P}}^H\mathbf{S}^{-1}\widetilde{\mathbf{P}})^{-1} = \mathbf{R} - \mathbf{R}\widetilde{\mathbf{P}}^H(\widetilde{\mathbf{P}}\mathbf{R}\widetilde{\mathbf{P}}^H + \mathbf{S})^{-1}\widetilde{\mathbf{P}}\mathbf{R}, \tag{4.15}$$

and the MSE $\triangleq \mathbb{E}\{\|\text{vec}(\mathbf{H}) - \text{vec}(\widehat{\mathbf{H}}_{\text{MMSE}})\|^2\}$ is

$$\text{MSE} = \text{tr}\left\{(\mathbf{R}^{-1} + \widetilde{\mathbf{P}}^H\mathbf{S}^{-1}\widetilde{\mathbf{P}})^{-1}\right\} \tag{4.16}$$

$$= \text{tr}\left\{\mathbf{R} - \mathbf{R}\widetilde{\mathbf{P}}^H(\widetilde{\mathbf{P}}\mathbf{R}\widetilde{\mathbf{P}}^H + \mathbf{S})^{-1}\widetilde{\mathbf{P}}\mathbf{R}\right\}. \tag{4.17}$$

The MMSE estimator of $\rho \triangleq \|\mathbf{H}\|_F^2$, with the observation $\mathbf{Y} = \mathbf{HP} + \mathbf{N}$ and training sequence $\mathbf{P}$, is

$$\hat{\rho}_{\text{MMSE}} = \text{vec}(\widehat{\mathbf{H}}_{\text{MMSE}})^H\text{vec}(\widehat{\mathbf{H}}_{\text{MMSE}}) + \text{tr}\{\mathbf{C}_{\text{MMSE}}\}. \tag{4.18}$$

The corresponding MSE is:

$$\text{MSE} = \text{tr}\{\mathbf{C}_{\text{MMSE}}(2\mathbf{R} - \mathbf{C}_{\text{MMSE}})\}. \tag{4.19}$$

We emphasize that the generic MMSE estimator in 4.14 is linear (affine), despite the fact that it has frequently been called the linear MMSE (LMMSE) estimator. This is accurate, but it may lead one to draw the false conclusion that there are possibly superior non-linear estimators. The LMMSE estimate is used in the situation of non-Gaussian fading and disturbance, and the MMSE estimator in 4.14 is also the maximum a posteriori (MAP) estimator of $\mathbf{H}$ [[20], Chapter 15.8]. (with known first and second order statistics, independent fading and disturbance, and possibly unknown types of distributions [[20], Chapter 12.3]).

It should be noted that the computation of 4.14 merely calls for the addition of a vector and the multiplication of vec($\mathbf{Y}$) with a matrix, both of which depend solely on the system statistics. As a result, the estimator's computational complexity is constrained.

4.2.3 Complex domain to real domain

Since $\widetilde{\mathbf{P}}, \mathbf{H}$, and $\mathbf{V}$ are complex in a communication system, we need to convert the system model into real domain before applying Gaussian BP.
More specifically, $vec(\mathbf{Y}) = \widetilde{\mathbf{P}} vec(\mathbf{H}) + vec(\mathbf{V})$ is equivalent to

$$\underbrace{\begin{bmatrix} \Re\{vec(\mathbf{Y})\} \\ \Im\{vec(\mathbf{Y})\} \end{bmatrix}}_{vec(\mathbf{Y})} = \underbrace{\begin{bmatrix} \Re\{\tilde{\mathbf{P}}\} & -\Im\{\tilde{\mathbf{P}}\} \\ \Im\{\tilde{\mathbf{P}}\} & \Re\{\tilde{\mathbf{P}}\} \end{bmatrix}}_{\tilde{\mathbf{P}}} \underbrace{\begin{bmatrix} \Re\{vec(\mathbf{H})\} \\ \Im\{vec(\mathbf{H})\} \end{bmatrix}}_{vec(\mathbf{H})} + \underbrace{\begin{bmatrix} \Re\{vec(\mathbf{V})\} \\ \Im\{vec(\mathbf{V})\} \end{bmatrix}}_{vec(\mathbf{V})} \tag{4.20}$$

After the procedure described above is completed, we are ready to apply Gaussian BP to the newly created vectors with $\mathbf{\Lambda} = \mathbf{S}^{-1}$, $\mathbf{\Xi} = \widetilde{\mathbf{P}}$, $\mathbf{\Sigma} = \mathbf{R}^{-1}$, $\boldsymbol{\xi} = \mathbf{0}$ and $\mathbf{u} = vec(\mathbf{Y})$ in Eq. 3.6.

4.2.4 Norm comparison And Selection of Threshold Value

In order to determine the presence or the absence of a moving person, at two consecutive instances t_1 and $t_1 + \delta t$, we calculate the difference of a metric (e.g. L_2 norm, Frobenius norm) of the matrix $\mathbf{H}$ of the two instances. If the difference exceeds a threshold value $\hat{\theta}$, we infer that motion occurred. Otherwise, we consider that there was no movement. The aforementioned are summarized below:

if $(\|\widehat{\mathbf{H}}\|^2_{t_1+\delta t} - \|\widehat{\mathbf{H}}\|^2_{t_1} \geq \hat{\theta})$ **then**

Presence of moving person

else

Absence of moving person

end if

Chapter 5

Numerical Results

In this Chapter, we present the experimental results of both *synchronous* and *asynchronous* variants of Gaussian Belief Propagation, used in the Linear Minimum Mean Square Error (LMMSE) Estimator. In particular, we provide simulations and numerical results studying the convergence of the algorithm, first for a general model and subsequently for the security model described in Chapter 4 . Furthermore, we compare the results of the MMSE estimator with the Least Squares (LS) Estimator and the original matrix $\mathbf{H}$ for different signal-to-noise ratios (SNRs).

In all the experiments, we consider *two* different kinds of schedules:

1. A *synchronous scheduling*,
2. An *i.i.d. asynchronous scheduling* [21] where the scheduling variables $\psi_i^{(l)}$ (eq. 3.24) are assumed i.i.d. Bernoulli random variables, with the same parameter p.

5.1 Linear Minimum Mean Square Error (LMMSE) Estimator

5.1.1 General Model

In the next experiments, we calculate a Gaussian vector's LMMSE, as defined in Section 3.3.

In all the experiments of this section, we assume that $\mathbf{W} = \mathbf{I}$, $\mathbf{R} = 0.1\mathbf{I}$.

Let

$$\mathbf{B_1} = \begin{bmatrix} -0.6505 & 0.3836 & 0 & 0 & 0 & 1.0566 & 0 & 0 \\ 0 & -0.5602 & -1.3901 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.2321 & -0.5892 & 0 & 0 & 0.2935 & 0 \\ 0 & 0 & 0 & 0.8142 & -0.2769 & 0 & 0 & 0 \\ -0.3459 & 0 & 0 & 0 & -0.1769 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0592 & -0.8274 & -0.4625 \\ -0.2197 & -0.4602 & 0 & -2.0636 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1647 & 0 & 0 & 0 & 0 & -1.6464 \end{bmatrix}, \tag{5.1}$$

$$\mathbf{B_2} = \begin{bmatrix} 72.3582 & 45.9351 & 91.6464 & 1.1806 & 50.4739 \\ 78.7834 & 69.2085 & 55.6571 & 77.0918 & 0.9754 \\ 20.2217 & 24.7729 & 4.9516 & 79.7035 & 46.5838 \\ 55.2631 & 59.2502 & 29.1595 & 89.2362 & 39.1887 \\ 82.0619 & 68.5507 & 77.0189 & 59.2605 & 99.3619 \end{bmatrix}, \tag{5.2}$$

$$\mathbf{B_3} = \begin{bmatrix} 54.3166 & 16.5279 & 20.4873 & 16.0691 & 57.2726 \\ 94.8873 & 61.0402 & 19.9851 & 88.8486 & 49.2473 \\ 9.1940 & 10.7729 & 20.9516 & 28.7557 & 8.6709 \\ 94.7030 & 13.0561 & 37.5078 & 18.6525 & 3.2260 \\ 17.4790 & 13.5482 & 53.8140 & 23.6038 & 17.0602 \end{bmatrix}, \tag{5.3}$$

the three matrices used for experiments 1,2 and 3 respectively. We plot the estimation of the per iteration expected value $\mathbb{E}[\|\mathbf{e}^{(l)}\|_2]$ using 10 independent experiments; at iteration (l), the error $\mathbf{e}$ is defined as

$$\mathbf{e}^{(l)} \triangleq \boldsymbol{\epsilon}^{(l)} - \hat{\mathbf{x}}, \tag{5.4}$$

where $\boldsymbol{\epsilon}^{(l)}$ is the vector that is constructed if we stack all *belief means*, according to Eq. 3.17, and $\hat{\mathbf{x}}$ the LMMSE estimator of $\mathbf{x}$ (Eq. 3.30). Finally, in Tables 5.1, 5.2, 5.3, we provide the parameters utilized in each experiment as well as the necessary convergence metrics.

Table 5.1: Experiment 1.

Scheduling	Probability of update	$\rho(\mathbf{A})$	$\rho(\mathbf{P}(\mathbf{A}-\mathbf{I})+\mathbf{I})$
synchronous	$p = 1$	0.55260	0.55260
i.i.d. asynchronous	$p = 0.55$	0.55260	0.75393

Table 5.2: Experiment 2.

Scheduling	Probability of update	$\rho(\mathbf{A})$	$\rho(\mathbf{P}(\mathbf{A}-\mathbf{I})+\mathbf{I})$
synchronous	$p = 1$	3.03970	3.03970
i.i.d. asynchronous	$p = 0.55$	3.03970	1.22184

Table 5.3: Experiment 3.

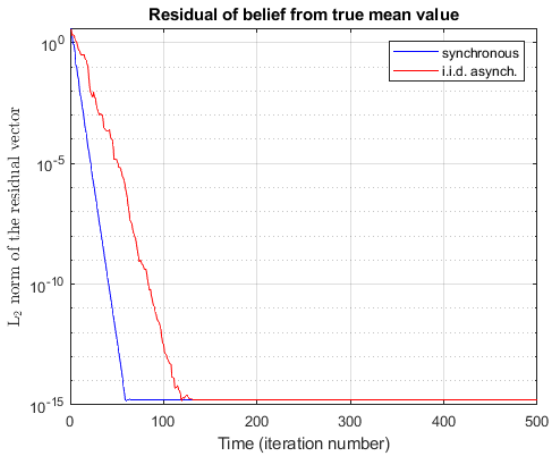
Scheduling	Probability of update	$\rho(\mathbf{A})$	$\rho(\mathbf{P}(\mathbf{A}-\mathbf{I})+\mathbf{I})$
synchronous	$p = 1$	1.81955	1.81955
i.i.d. asynchronous	$p = 0.55$	1.81955	0.88035

In Figs. 5.1a, 5.1b, 5.1c and in Tables 5.1, 5.2, 5.3, we show the convergence plots (similarly to the previous case, we plot the mean $\mathbb{E}[\|\boldsymbol{\epsilon}^{(l)} - \hat{\mathbf{x}}\|_2]$ at every iteration) along the parameters that were used in each experiment.

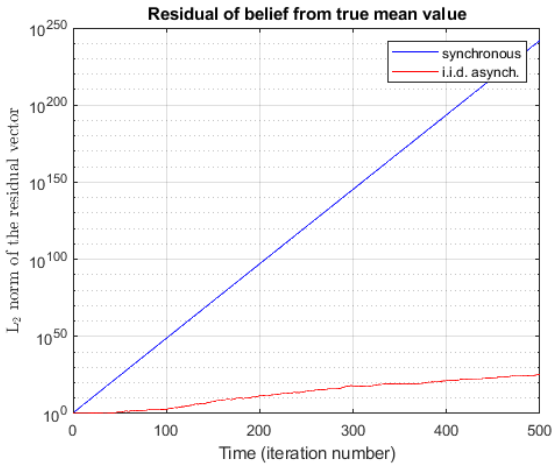
First of all, in experiment 1, we notice that $\rho(\mathbf{A}) = 0.5520 < 1$; as a result we expect Gaussian Belief Propagation to converge under synchronous scheduling, something we can verify from the plots. In addition, we see that $\rho(\mathbf{P}(\mathbf{A}-\mathbf{I})+\mathbf{I}) < 1$ for the asynchronous scheduling, something that -as we can derive from the plots- implies convergence.

In experiment 2, both $\rho(\mathbf{A}) = 3.03970 > 1$ and $\rho(\mathbf{P}(\mathbf{A}-\mathbf{I})+\mathbf{I}) = 1.22184 > 1$. Thus, we expect both synchronous and asynchronous scheduling to diverge, as is evidenced by the relevant figure.

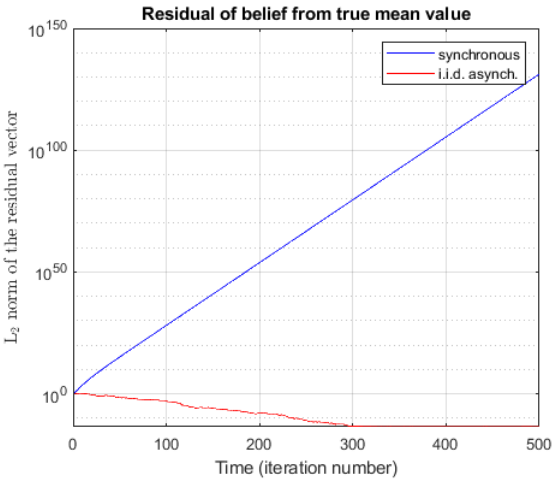
In experiment 3, $\rho(\mathbf{A}) = 1.81955 > 1$ while $\rho(\mathbf{P}(\mathbf{A}-\mathbf{I})+\mathbf{I}) = 0.88035 < 1$. In this case, Gaussian Belief Propagation under synchronous scheduling diverges, while asynchronous scheduling achieves convergence. Therefore, this experiment stresses the necessity of the asynchronous variant of the algorithm, as it is able to converge, contrary to the synchronous case.



(a) Experiment 1



(b) Experiment 2



(c) Experiment 3

Figure 5.1: Convergence of Gaussian Belief Propagation for calculating the LMMSE estimator of a Gaussian vector under different schedulings.

5.1.2 Security Model

In this section, we implement the security model described in Chapter 4. We consider $M = 3$ antennas on the transmitter, $N = 6$ single-antenna receivers and $M_{T_x} = M = 3$ training vectors. We also set the following: distance vector $\mathbf{d}_k = [0.5 \quad 1 \quad 2 \quad 3 \quad 4 \quad 6]m$, wavelength $\lambda = 0.125m$ (assuming 2.4 GHz Wi-Fi), reference distance $d_0 = 1m$, path-loss exponent $u_k = 3$, oversampling parameter $L = 10$ and temperature $T = 300K$.

In figure 5.2, and considering $\mathbf{P} = (0.85)\mathbf{I}$ for the asynchronous case, we notice that both the synchronous and the asynchronous variants of the MMSE estimator converge, since $\rho(\mathbf{A}) \approx 10^{-38} \ll 1$ and $\rho(\mathbf{P}(\mathbf{A}-\mathbf{I})+\mathbf{I}) = 0.15 < 1$ respectively. It is also important to note that the L_2 norm of the residual vector converges from the first iteration in the synchronous variant.

For the experiments below, the signal-to-noise ratio (SNR) and the SNR expressed in dB are:

$$SNR = \frac{\|\mathbf{P}\|^2 \cdot L}{\sigma_n^2} \tag{5.5}$$

$$SNR_{dB} = 10 \log_{10} SNR \tag{5.6}$$

Let $R_i, i \in \{1, .., N\}$ the receiver whose distance from the transmitter is d_i. The results from the metrics differences are shown in Tables 5.4, 5.5, 5.6, 5.7, 5.8, 5.9. Moreover, the presence of a person in front of an estimator is modeled by multiplying the specific row of the channel row by a constant $c \in \mathbb{R}$. For our experiments, we set $c = 100$.

For Experiments 1 and 2, the results of the motion detection are shown in Tables 5.5,5.6,5.8,5.9. We assume that for $t = t_1$ no motion is being detected, whereas for $t = t_1 + \delta t$, the sensor R_i has detected the presence of a moving person.

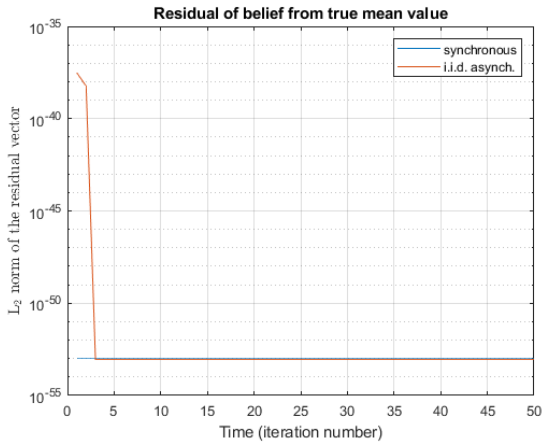


Figure 5.2: Synchronous vs asynchronous implementation of the MMSE estimator on security model

Table 5.4: Experiment 1 (Synchronous case, no motion)

SNR (dB)	$\|\hat{\mathbf{H}}_{\mathbf{MMSE}}\|^2_{F,t_1+\delta t} - \|\hat{\mathbf{H}}_{\mathbf{MMSE}}\|^2_{F,t_1}$	$\|\hat{\mathbf{H}}_{\mathbf{LS}}\|^2_{F,t_1+\delta t} - \|\hat{\mathbf{H}}_{\mathbf{LS}}\|^2_{F,t_1}$
7.30	2.4061	5.3245
11.28	3.0929	4.8212
14.29	2.2710	2.8705
17.30	0.4824	0.5442

Table 5.5: Experiment 1 (Synchronous MMSE, with motion detection) $\|\widehat{\mathbf{H}}_{\mathbf{MMSE}}\|^2_{F,t_1+\delta t} - \|\widehat{\mathbf{H}}_{\mathbf{MMSE}}\|^2_{F,t_1}$

SNR (dB)	R_1	R_2	R_3	R_4	R_5	R_6
7.30	23.8658	24.1930	22.3473	26.0661	20.3075	27.1186
11.28	34.6271	36.8589	36.1883	35.0369	30.8352	33.7816
14.29	33.2661	34.9766	40.5446	37.9804	42.8286	40.8749
17.30	8.6584	7.0956	7.7392	7.5230	7.4003	6.1309

Table 5.6: Experiment 1 (LS, with motion detection) $\|\widehat{\mathbf{H}}_{\mathbf{LS}}\|^2_{F,t_1+\delta t} - \|\widehat{\mathbf{H}}_{\mathbf{LS}}\|^2_{F,t_1}$

SNR (dB)	R_1	R_2	R_3	R_4	R_5	R_6
7.30	62.7332	63.5934	58.7418	68.5169	53.3799	71.2838
11.28	53.9766	57.4554	56.4101	54.6153	48.0658	52.6586
14.29	42.0469	44.2090	51.2467	48.0056	54.1336	51.6641
17.30	9.7677	8.0047	8.7307	8.4869	8.3485	6.9164

Table 5.7: Experiment 2 (Asynchronous case, no motion)

SNR (dB)	$\|\widehat{\mathbf{H}}_{\mathbf{MMSE}}\|^2_{F,t_1+\delta t} - \|\widehat{\mathbf{H}}_{\mathbf{MMSE}}\|^2_{F,t_1}$	$\|\widehat{\mathbf{H}}_{\mathbf{LS}}\|^2_{F,t_1+\delta t} - \|\widehat{\mathbf{H}}_{\mathbf{LS}}\|^2_{F,t_1}$
7.30	1.6152	2.5030
11.28	2.4085	3.7544
14.29	1.7733	2.2414
17.30	0.8436	0.9773

Table 5.8: Experiment 2 (Asynchronous MMSE, with motion detection) $\|\widehat{\mathbf{H}}_{\mathbf{MMSE}}\|^2_{F,t_1+\delta t} - \|\widehat{\mathbf{H}}_{\mathbf{MMSE}}\|^2_{F,t_1}$

SNR (dB)	R_1	R_2	R_3	R_4	R_5	R_6
7.30	19.9685	19.5323	17.4773	21.7415	15.6598	18.1679
11.28	15.2822	13.6825	16.3492	16.2191	15.6532	21.3002
14.29	17.4205	19.2783	17.2219	19.0013	18.0619	16.5061
17.30	14.3674	12.4379	14.3874	15.8892	13.3658	14.4374

Table 5.9: Experiment 2 (LS, with motion detection) $\|\widehat{\mathbf{H}}_{\mathbf{LS}}\|^2_{F,t_1+\delta t} - \|\widehat{\mathbf{H}}_{\mathbf{LS}}\|^2_{F,t_1}$

SNR (dB)	R_1	R_2	R_3	R_4	R_5	R_6
7.30	52.4890	51.3425	45.9406	57.1496	41.1631	47.7558
11.28	23.8219	21.3282	25.4850	25.2823	24.4002	33.2027
14.29	22.0188	24.3670	21.7678	24.0168	22.8294	20.8630
17.30	17.4893	15.3126	17.5119	19.2060	16.3594	17.5683

In our last experiment, depicted in Table 5.10 we present the squared Frobenius norms of matrix $\mathbf{H}$, of the centralized MMSE estimator and of the distributed MMSE estimator (synch. and asynch) and for the LS estimator for $SNR = 14.29dB$:

Table 5.10: Experiment 3

$\|\mathbf{H}\|^2_F$	$\|\widehat{\mathbf{H}}_{MMSE,cent.,s}\|^2_F$	$\|\widehat{\mathbf{H}}_{MMSE_{d.,a.}}\|^2_F$	$\|\widehat{\mathbf{H}}_{MMSE_{d.,s.}}\|^2_F$	$\|\widehat{\mathbf{H}}_{LS}\|^2_F$
0.2159	0.2158	0.2158	0.2158	0.2160

We conclude that both estimators are able to accurately identify whether or not a moving person is present or not by the difference of the squared Frobenius norm at time $t = t_1 + \delta t$ compared to time t_1. More specifically, for the experiments where mobility occurs at $t = t_1 + \delta t$ there is at least one order of magnitude change compared to $t = t_1$ when no motion has yet occurred. We also deduce that for higher SNR values, the estimation is more accurate compared to lower SNRs, as the signal's power is stronger and therefore more prominent compared to the system noise. Finally, the value of the distributed MMSE estimator is equal or almost equal to that of the centralized MMSE estimator, in both synchronous and asynchronous schedulings.

Chapter 6

Conclusions and Future Work

In this work, our goal was to implement a privacy-non intrusive security application exploiting the benefits of asynchronous and distributed inference. It involved distributed wireless terminals/sensors in an indoor space that aim to measure the wireless channel and detect, between consecutive measurements, the presence (or absence) of a moving person, without reverting to cameras, microphones or other means associated with privacy intrusion.

After presenting the convergence conditions of distributed inference using Gaussian Belief Propagation and the way GBP can be utilized for solving an MMSE estimation problem, we provided the theoretical framework of the centralized LS and MMSE estimators that were used in our security problem.

Finally, according to the numerical results of our security system, which assumed WiFi carrier frequency of 2.4GHz and Rayleigh fading channel, we can conclude that mobility can be detected with significant changes in the channel estimate metrics in both synchronous and asynchronous variants of the GBP using MMSE and LS estimators. Mobility is assumed only if the difference in the norm of the channel estimate between

two consecutive measurements exceeds a carefully selected threshold value.

Some interesting future directions of this work could include the existence of additional sensors for the MMSE channel estimation, other WiFi carrier frequencies (e.g. 5GHz) and other channel distributions, such as the Rician fading.

Bibliography

[1] J. Du, S. Ma, Y.-C. Wu, S. Kar, and J. M. F. Moura, "Convergence analysis of distributed inference with vector-valued gaussian belief propagation," *Journal of Machine Learning Research*, vol. 18, no. 172, pp. 1–38, Apr. 2018.

[2] Y. Weiss and W. T. Freeman, "On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs," *IEEE Trans. Inf. Theor.*, vol. 47, no. 2, pp. 736–744, Sep. 2001.

[3] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[4] D. Shah, "Algorithms for inference—mit course no. 6.438," Cambridge MA, 2014, MIT OpenCourseWare. [Online]. Available: https://ocw.mit.edu/courses/6-438-algorithms-for-inference-fall-2014/

[5] 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, 2001.

[6] B. Li and Y.-C. Wu, "Convergence analysis of gaussian belief propagation under high-order factorization and asynchronous scheduling," *IEEE Transactions on Signal Processing*, vol. 67, no. 11, pp. 2884–2897, 2019.

[7] V. Papageorgiou, A. Nichoritis, P. Vasilakopoulos, G. Vougioukas, and A. Bletsas, "Towards ambiently powered internet-of-things-that-think with asynchronous princi-

ples," School of Electrical and Computer Engineering, Technical University of Crete, Tech. Rep., 2022.

[8] ——, "Towards ambiently powered inference on wireless sensor networks: Asynchrony is the key!" in *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2021, pp. 458–465.

[9] C. D. Meyer, *Matrix analysis and applied linear algebra*. Siam, 2000, vol. 71.

[10] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. USA: Prentice-Hall, Inc., 1989.

[11] O. Teke and P. P. Vaidyanathan, "Random node-asynchronous graph computations: Novel opportunities for discrete-time state-space recursions," *IEEE Signal Processing Magazine*, vol. 37, no. 6, pp. 64–73, 2020.

[12] ——, "Randomized asynchronous recursions with a sinusoidal input," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2019, pp. 1491–1495.

[13] ——, "Random node-asynchronous updates on graphs," *IEEE Transactions on Signal Processing*, vol. 67, no. 11, pp. 2794–2809, 2019.

[14] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[15] B. C. Levy, *Principles of Signal Detection and Parameter Estimation*. New York: Springer, 2008.

[16] M. G. Amin, *Through-the-Wall Radar Imaging*. CRC Press, 2010.

[17] M. Biguesh and A. Gershman, "Training-based mimo channel estimation: a study of estimator tradeoffs and optimal training signals," *IEEE Transactions on Signal Processing*, vol. 54, no. 3, pp. 884–893, 2006.

[18] E. Bjornson and B. Ottersten, "A framework for training-based estimation in arbitrarily correlated rician mimo channels with rician disturbance," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1807–1820, 2010.

[19] A. Goldsmith, *Wireless Communications*. New York, NY, USA: Cambridge University Press, 2005.

[20] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Englewood Cliffs, N.J.: Prentice-Hall, 1993.

[21] B. Li and Y.-C. Wu, "Convergence analysis of gaussian belief propagation under high-order factorization and asynchronous scheduling," *IEEE Trans. Signal Processing*, vol. 67, no. 11, pp. 2884–2897, Jun. 2019.