Bachelor thesis

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Control Engineering

Wireless Channel Parameter Estimation using Artificial Neural Networks

Vítek Udatný

Supervisor: Ing. Rostislav Karásek
Field of study: Cybernetics and Robotics
May 2021

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Udatný Vítek**	Personal ID number:	**474432**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Control Engineering**		
Study program:	**Cybernetics and Robotics**		

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

Wireless Channel Parameter Estimation using Artificial Neural Networks

Bachelor’s thesis title in Czech:

Odhad parametrů rádiového kanálu s využitím umělých neuronových sítí

Guidelines:

The modern positioning algorithms aim for precise position estimation in environments that are challenging for the classical methods. One of the most challenging problems is to estimate position indoors. Currently, the most promising approach to this challenge is a technique known as multipath assisted positioning. The multipath assisted positioning approach's accuracy is restricted by the precision of estimated wireless channel parameters. The state-of-the-art methods' computational complexity is not allowing real-time processing required for positioning.

One promising approach to solve the problem mentioned above is to utilize recent advances in machine learning algorithms and artificial neural network architectures. The thesis aims to study and apply relevant machine learning techniques to estimate the wireless channel parameters and compare this approach with the state-of-the-art method.

- Study the state-of-the-art methods of wireless channel parameter estimation.
- Program a simulator for generating channel measurement realizations according to the multipath channel model with dense multipath components.
- Propose an artificial neural network architecture suitable for wireless channel parameter estimation.
- Train the proposed artificial neural network using the simulated data.
- Compare the performance of the proposed method with the theoretical limits and the state-of-the-art maximum likelihood estimator performance.

Bibliography / sources:

[1] B. H. Fleury, M. Tschudin, R. Heddergott, D. Dahlhaus and K. Ingeman Pedersen, "Channel parameter estimation in mobile radio environments using the SAGE algorithm," in IEEE Journal on Selected Areas in Communications, vol. 17, no. 3, pp. 434-450, March 1999

[2] A. Richter, "Estimation of Radio Channel Parameters: Models and Algorithms " Ph.D. dissertation, TU Ilmenau, May 2005

[3] Y. -S. Hsiao, M. Yang and H. -S. Kim, "Super-Resolution Time-of-Arrival Estimation using Neural Networks," 2020 28th European Signal Processing Conference (EUSIPCO), Amsterdam, 2021

[4] P. Stoica, A. Jakobsson, and J. Li, "Cisoid Parameter Estimation in the Colored Noise Case: Asymptotic Cramér-Rao Bound, Maximum Likelihood, and Nonlinear Least-Squares," IEEE Transactions on Signal Processing, vol. 45, no. 8, Aug. 1997.

[5] J. M. Francos and B. Friedlander, "Bounds for Estimation of Complex Exponentials in Unknown Colored Noise," IEEE Transactions on Signal Processing, vol. 43, no. 9, Sep. 1995.

[6] P. A. Bello, "Characterization of Randomly Time-Variant Linear Channels," IEEE Transactions on Communications, vol. 11, no. 4, 1963.

Name and workplace of bachelor’s thesis supervisor:

Ing. Rostislav Karásek, Department of Electromagnetic Field, FEE

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

prof. Ing. Pavel Pechač, Ph.D., Department of Electromagnetic Field, FEE

Date of bachelor’s thesis assignment: __________ Deadline for bachelor thesis submission: __________

Assignment valid until:
by the end of summer semester 2021/2022

Ing. Rostislav Karásek Supervisor’s signature	prof. Ing. Michael Šebek, DrSc. Head of department’s signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

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

.
Date of assignment receipt

Student’s signature

Acknowledgements

I would like to thank my supervisor Ing. Rostislav Karásek for his guidance, most welcomed advises, willingness and patience during our consultations. Last but not least, I wish to thank my family, girlfriend and friends for their support throughout my studies.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

In Prague, May 2021

Abstract

The main goal of this thesis is to explore opportunities the artificial neural networks could offer to the field of wireless parameter estimation. We theoretically and algebraically describe complex channel model that consists of multi-path components, dense multi-path components and additive white Gaussian noise. Channel impulse response simulator capable of both correctly providing us artificial channel impulse responses and respecting limits of real measurement equipment was implemented. We focus on estimating time of arrival and parameters of dense multi-path components. SAGE algorithm is used as a state-of-the-art estimation performance reference. We propose a method based on a two stage artificial neural network. The first stage roughly estimates delay of multi-path components and power delay profile. Then, the second stage provides fine estimation of delay based on the rough estimates. Results are compared with the estimates of reference SAGE algorithm and Cramér-Rao lower bound.

Keywords: signal processing, wireless channel parameter estimation, multi-path components, dense multi-path components, SAGE, Cramér-Rao lower bound, time of arrival, artificial neural network

Supervisor: Ing. Rostislav Karásek
Czech Technical University in Prague,
Department of Electromagnetic Field
and German Aerospace Center (DLR),
Institut für Kommunikation und
Navigation

Abstrakt

Hlavním cílem této práce je prozkoumat možnosti umělých neuronových sítí na poli odhadování parametrů rádiového kanálu. Teoreticky a algebraicky jsem popsal model bezdrátového kanálu, vytvořil simulátor impulsových odezev, který produkuje teoreticky přesné vzorky, ale zároveň respektuje omezení reálných měřících zařízení. Uvedený a implementovaný SAGE algoritmus slouží jako reference odhadů. Představil jsem metodu založenou na dvouúrovňovém zpracování signálu pomocí neuronových sítích, které v první řadě hrubě odhadují časové zpoždění komponent vícecestného šíření spolu s výkonovými profily šumu tvořeného za hlavními vrcholy. Tyto odhady jsou následně použity spolu s originální impulzovou odezvou jako vstup finální neuronové sítě, která precizně odhaduje zpoždění komponent vícecestného šíření. Výsledky byly porovnány s odhady referenčního SAGE algoritmu a teoretickou spodní mezí Cramér-Rao lower bound.

Klíčová slova: zpracování signálů, odhad parametrů rádiového kanálu, vícecestné šíření, SAGE, Cramér-Rao spodní limit, časové zpoždění, umělé neurononové sítě

Překlad názvu: Odhad parametrů rádiového kanálu s využitím umělých neuronových sítí

Contents

Part I
Theoretical part

Part II
Practical part

Appendices

Figures

Tables

Chapter 1

Introduction

1.1 Motivation

The modern positioning algorithms aim for precise position estimation in environments that are challenging for the classical methods. One of the most challenging problems is to estimate position indoors. Currently, the most promising approach to this challenge is a technique known as multi-path assisted positioning. The multi-path assisted positioning approach's accuracy is restricted by the precision of estimated wireless channel parameters. Computational complexity of state-of-the-art methods are not allowing real-time processing required for positioning.

One promising approach to solve the problem mentioned above is to utilize recent advances in machine learning algorithms and artificial neural network architectures.

1.2 Outline

The thesis aims to study and apply relevant machine learning techniques to estimate the wireless channel parameters and compare this approach with the state-of-the-art method and theoretical lower bound of parameter estimator precision.

For purposes of our theoretical research it is necessary to have datasets containing channel impulse responses (CIRs) generated by our model given set of parameters and their precise ground truth labels. It is not possible to obtain such samples by real measurement because they are affected by uncontrollable random processes. This implies need of a channel impulse response simulator capable of both correctly providing synthetic artificial channel measurements and respecting limits of measurement equipment in order to obtain realistic signals.

We have developed CIR simulator that is based on derived algebraic complex channel model that consists of multi-path components (MPCs), dense multi-path components (DMCs) and additive white Gaussian noise (AWGN). DMCs [21] consist of high number of scattered propagation paths associated with particular MPC which can not be observed within resolution of measured equipment and thus is modeled as a stochastic random signal with characteristic power delay profile (PDP) [5], [20]. DMCs are observed as increased noise following MPCs and exponentially decaying in time. These components are often ignored, causing the performance of the channel parameter estimator to be suboptimal. Channel analysis is based on channel measurements where first a signal is transmitted. The signal propagates through the environment. Then, the received signal is distorted by the environment. This distortion can be described by a channel impulse response characterizing the wireless channel. Measurements contain finite amount of information. Channel measurement equipment is always of a finite bandwidth [22] but we could recover the information with infinite resolution under certain conditions [2], e.g., sufficient separation of delays. Unfortunately in real world scenarios the received signal is corrupted by noise and so we receive incomplete data [22],[14] from witch we can not recover the information with perfectly.

The model has to satisfy at first glance two contradicting conditions on the best level possible. It has to flawlessly balance in between correct description of a wireless channel on the one hand and respect level of observability complexity that can be achieved on the other hand. It is important to respect these limitations in order to keep the research work practical.

We focus on estimating time of arrival (ToA) of MPCs and power delay profiles (PDPs) of DMCs. Complex amplitudes and phase shifts can be estimated using best linear unbiased estimator (BLUE) [22], [14]. Phase shifts are considered to be distributed uniformly throughout our work and therefore carry no information about ToAs. Once ToAs are estimated, it is elementary to obtain the amplitudes by simply reading the value from the signal.

Theoretical limit on ToAs estimation broadly accepted and used [22],[8], [25] is the Cramér-Rao Lower Bound (CRLB) first introduced in [3] and is used throughout the thesis to evaluate estimator's performance.

SAGE algorithm [6] as a state-of-the-art maximum likelihood estimator is presented and implemented. It serve us as a performance reference algorithm. A few drawbacks come with SAGE algorithm. Starting with its high computational complexity, need of apriory knowledge of number of MPCs and providing initial estimates. The main problem is however that SAGE at is original form [6] ignores important component of the signal that are DMCs, this results in a significant estimation performance drop. Additional methods that address some of the mentioned drawbacks of SAGE algorithms were proposed [13].

Inspired by [12] we propose method which is based on two-level processing using artificial neural networks (ANNs) that first roughly estimates ToAs of MPCs and PDP of DMCs using two ANNs and feed cropped (focused) windows of these estimates as inputs into the second-level ANN which fine estimates ToAs of MPCs. The performance of our methods is compared to CRLB and to SAGE algorithm in terms of root mean square error (RMSE).

Part I

Theoretical part

Chapter 2

Wireless Channel Model

In this section the focus will be on description of mathematical framework of an observation of wireless channel. First basic concepts and components will be introduced following the algebraic model which is core of our work and is being used throughout the entire thesis.

2.1 Channel Measurement

In order to investigate channel parameters we need to obtain channel measurement. Although we don't work with any real channel measurement data in our work, we find it correct and useful for our readers to briefly describe the measurement process. For purposes of our theoretical research it is necessary to have datasetscontaining CIRs of desired parameters and their precise ground truth labels.It is not possible to obtain such samples by real measurement.

To work with a channel we first need to identify its parameters. Identification of the channel parameters is usually done by measuring the channel impulse response. It is very difficult process and there are a lot of approaches which are summarized in [19] applicable. We could use either time-domain measurement techniques or frequency-domain measurement techniques while considering the time-frequency duality concept [1]. The techniques could be either passive or active, require various laboratory equipment and provide diverse quality results. It is necessary to use high-end instruments that consists of top-quality materials and parts and deploy robust and complex signal processing hardware architectures and algorithms to obtain proper results.

Majority of techniques are active and require to first send a reference sounding signal into the environment and then listen for its response, they are also referred as channel sounding techniques. State-of-the-art radio channel sounding techniques are described in [24].

We would particularly mention channel impulse response estimation done by means of frequency-domain wide-band sounding approach which utilizes Vector Network Analyzer (VNA) [28].

2.2 Basic concepts and components

In following section we will introduce the basic concepts that we work with when constructing the wireless channel model. It is important for the reader to gain all necessary information and background that we utilize in later section when deriving algebraic channel model.

2.2.1 AWGN channel

The simplest form of a channel in our work is additive white Gaussian noise channel where there is no contribution of transmitted signal to the received one. We obtain a sample of AWGN channel by pure measuring the channel and not transmitting anything.

World is full of electromagnetic noise which is an effect of many random processes produced either by humans of by nature itself. We consider the noise to have equal power among frequency spectrum. Noise parameters are determined by the environment of signal propagation. Parameters of electromagnetic noise are for example strongly dependant on temperature. In our work broadly accepted and used concept of AWGN is used.

2.2.2 Definition of a ray

The smallest entity in wireless channel that we utilize in our work is a ray.

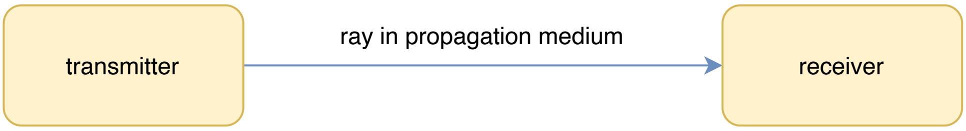

Figure 2.1: Definition of a ray.

It is known that using ray-optical modelling it is possible to approximately model complex wave propagating phenomena in continuum like reflection, diffraction, and scattering. Ray-optical modelling comes with rays possessing several parameters that describe their propagation through given environment. Such parameters might be for example transmit/receive azimuth, transmit/receive elevation which are spatial angles of ray departure and arrival. For the purposes of our work the main parameter of interest is ToA τ.

2.2.3 Definition of a propagation path

Using the elementary ray model is great base concept for many applications but due to noisy and corrupted received signal is not applicable in real world scenarios. We make use of model of propagation path that is essentially a cluster of rays that are close to each other. The very same parametrization is used while describing both rays and propagation paths - the parameter ToA τ is applicable while describing the clustered rays.

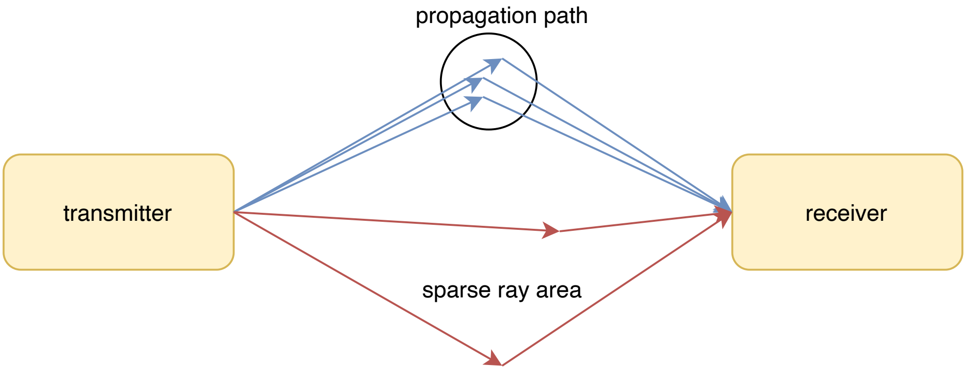

Figure 2.2: Definition of a propagation path.

In figure 2.2 rays represented as arrows are being propagated through certain medium. They interact with surrounding environment and object located in the environment while following geometric optics principles and phenomena such as reflection, diffraction and scattering [27]. Rays following same spatial and temporal propagation paths (eg.: reflected by the same objects at same angles) will likely have their propagation paths affected in a similar way and thus form a dominant propagation paths. Such rays are colored in blue. Strongly diffracted and scattered rays will in general be part of sparse ray areas and will not be part of any dominant path clusters. Such rays are colored in red.

2.2.4 Definition of multi-path propagation

Multi-path propagation is phenomena in wireless channel signal propagation where the signal reaches receiver by two and more paths. Because the signal went through various transformations while travelling various paths it possesses various parameters. We say that received signal is then composed of multi-path components (MPC). Figure 2.3 demonstrates this case. Their presence brings interference which could be either constructive and destructive. Each component brings new information to our measurement and we can utilize it to improve our performance of algorithms. Unfortunately in case of destructive interference the signal might be losing on its power and the carried information might be fading out. It is important to be aware of these possible situations and adjust our algorithms in order to achieve best performance.

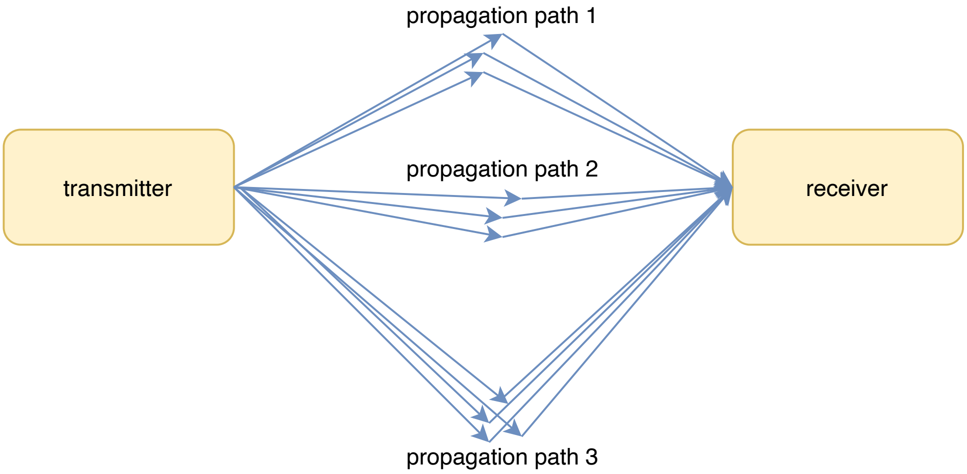

Figure 2.3: Definition of a multi-path propagation.

It can be in figure 2.3 seen that the received signal has reached receiver by three separate paths. The received signal consists of three multi-path components.

2.2.5 Definition of dense multi-path components

When transmitted signal reaches some propagation media crossings, parts of the signal can either continue travelling into the next medium or they are reflect or scattered. It is much easier for the signal to be scattered rather than reflected [22]. Reflection requires reflecting surface, sufficiently large object and correct combination position of next reflector or receiver. Despite the fact that it is hard for the signal to be reflected in such a way that it reaches the receiver in the end, reflected signals usually dominates the transmission [22]. On the other hand, scattered signals do not carry significant amount of energy compared to the reflected ones but because of their large number, the final contribution to the received signal can not be ignored. Dense multi-path components are accompanying the components produced by dominant propagation paths because a lot of scattering happens as a side effect of propagation of the dominant components.

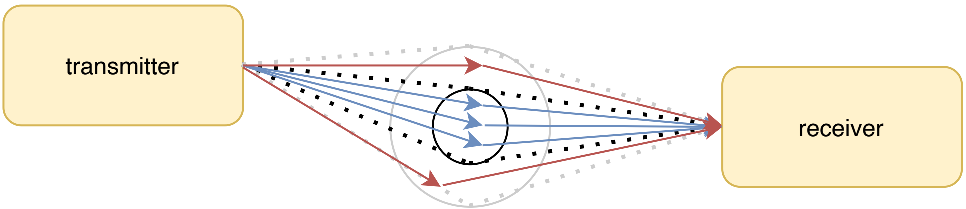

Figure 2.4: Definition of a dense multi-path components.

In figure 2.4 blue arrows are representing rays composing the dominant signal component (the MPC) which is reflected in the high reflection area (inner black circle) and red arrows follows the same propagation paths but fall into are of worse reflecting conditions (outer grey circle) and are not reflected such cleanly as the blue arrows.

Big amount of low power DMCs can't be observed and we can't receive complete information from the measured data. They are essentially detected as increased level of noise following dominant components that decays in time domain. DMCs are thus modelled as a stochastic process with normal distribution with exponentially decaying power in time [21].

2.3 Algebraic Model

In this section the algebraic model for our wireless channel is derived and defined. More specifically we provide channel impulse response algebraic description combining both deterministic and stochastic components.

2.3.1 Complex function

The channel impulse response is a complex valued function thus the core of our algebraic model is a complex function $f : \mathbb{R} \rightarrow \mathbb{C}$

$$y = f(x) \tag{2.1}$$

where $x \in \mathbb{R}$ and $y \in \mathbb{C}$. We won't go through elementary introduction to complex numbers, we follow broadly accepted principles which could be found in [23].

2.3.2 Impulse representation

Transmission of a single impulse in time τ could be described using

$$x(t) = \delta(t - \tau) \tag{2.2}$$

where $x(t) : \mathbb{R} \rightarrow \mathbb{R}$ and δ is Dirac impulse defined as

$$\delta(x) \begin{cases} \infty & x = 0 \\ 0 & x \neq 0 \end{cases} \tag{2.3}$$

which fulfills

$$\int_{-\infty}^{\infty} \delta(x)dx = 1. \tag{2.4}$$

Proper definition and description of Dirac impulse in [18].

Frequency domain image of Dirac impulse using Fourier transform is

$$\hat{\delta}(f) = F\{\delta(t - \tau)\} = \int_{-\infty}^{\infty} \delta(x)e^{-i2\pi ft}dt = e^{-i2\pi f\tau}. \tag{2.5}$$

2.3.3 Multi-path component model

In our work we will utilize generally accepted multi-path component model use for example in [24],[25] and [7].

Single component model in time domain with continuous time would be

$$h(t) = \alpha e^{i\theta}\delta(t-\tau) \tag{2.6}$$

where t is time, τ is time delay of the component, α is amplitude of the component and θ is phase of the component. Frequency domain with continuous frequency model is defined as

$$H(f) = \alpha e^{i\theta}e^{-i2\pi f\tau}. \tag{2.7}$$

where f is frequency and other parameters are very same as in (2.6) Transfer to a model which consists of multiple components is done by simply summing components together, for time-domain continuous time model the equation is

$$h(t) = \sum_{n=0}^{N} \alpha_n e^{i\theta_n}\delta(t-\tau_n) \tag{2.8}$$

where parameters $\boldsymbol{\alpha}$, $\boldsymbol{\theta}$ and $\boldsymbol{\tau}$ are vectors of amplitude, phase and time delay such as in (2.6) and N is number of components.

Corresponding definition for frequency domain continuous frequency model would be

$$H(f) = \sum_{n=0}^{N} \boldsymbol{\alpha}_n e^{i\boldsymbol{\theta}_n}e^{-i2\pi f\boldsymbol{\tau_n}}. \tag{2.9}$$

where f is frequency and other parameters are very same as in (2.8).

Descrete time/frequency domain forms of equations (2.6), (2.7), (2.8) and (2.9) are

$$h^1[k_t] = \alpha e^{i\theta}\delta[k_t-\tau], \tag{2.10}$$

$$H^1[k_f] = \alpha e^{i\theta}e^{-i2\pi k_f\tau}, \tag{2.11}$$

$$h[k_t] = \sum_{n=0}^{N} \alpha_n e^{i\theta_n}\delta[k_t-\tau_n], \tag{2.12}$$

$$H[k_f] = \sum_{n=0}^{N} \alpha_n e^{i\theta_n}e^{-i2\pi k_f\tau_n} \tag{2.13}$$

respectively, preserving meaning of parameters.

2.3.4 Noise model

Channel which consists of AWGN is in general modelled as

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{w} \tag{2.14}$$

where $\boldsymbol{y}$ is channel observation, $\boldsymbol{x}$ deterministic channel component and $\boldsymbol{w}$ is the AWGN which is a identically and independently distributed random vector. All w_i (2.15) are drawn from the same distribution and are mutually independent.

$$w_i \sim \mathcal{N}(\mu, \sigma^2) \tag{2.15}$$

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\cdot(\frac{x-\mu}{\sigma})^2} \tag{2.16}$$

AWGN is based on Normal distribution with zero mean μ and standard deviation σ (2.15), (2.16).

Total energy and average power [18] of a signal $x(t)$ can be calculated using continuous-time equations

$$E_x = \int_{-\infty}^{\infty} |x(t)|^2 dt, \tag{2.17}$$

$$P_x = \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} |x(t)|^2 dt \tag{2.18}$$

and their discrete-time forms are

$$E_x = \sum_{k=-\infty}^{\infty} |x[k]|^2, \tag{2.19}$$

$$P_x = \lim_{K\to\infty} \frac{1}{2K} \sum_{k=-\infty}^{\infty} |x[k]|^2 \tag{2.20}$$

This is however only case for $W_i \in \mathbb{R}$ and our signal - channel impulse response is a complex signal.

Let us define

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{z} \tag{2.21}$$

which is based on (2.14) unless the $z_i \in \mathbb{C}$ where $z_i = z_i^{(1)} + iz_i^{(2)}$ and $z_i^{(1)} + iz_i^{(2)}$ are independent random variables of (2.15). Then the distribution of Z_i is called to be Complex circular Normal distribution. Equations for signal energy and power (2.17), (2.18), (2.17) and (2.20) still hold true for (2.21) however it is important to remember that the noise is composed of its real and imaginary component which together form a signal of desired power and appropriately adjust variances σ^2.

2.3.5 Dense multi-path components model

Each DMC is associated with particular MPC. It has been mentioned that the DMC is presented as a increased noise after the associated MPC, which is exponentially decaying in time [21].

DMC signal is a stochastic complex signal with zero-mean of Complex circular Normal distribution and it is characterized it by its power delay profile (PDP) computed using (2.18), (2.20).

Time-domain model has been used in [22], [5] and [20]. It is essentially stochastic approximation. This proposed model provides reasonably good estimation of variance (power) as a function of time of dense multi-path components.

$$\psi_h(t) = \mathrm{E}\{|h(t)|^2\} = \begin{cases} 0 & \text{if } t < \tau_c \\ \beta \cdot \frac{1}{2} & \text{if } t = \tau_c \\ \beta \cdot e^{-B(t-\tau_c)} & \text{if } t > \tau_c \end{cases} \tag{2.22}$$

where t is time, τ_c is time delay of associated MPC, β is maximum variance (power) of DMC and B is decay speed. The decay speed B could be bandwidth-based on the coherence bandwidth which comes from Rayleigh distribution of dense multi-path components such as in [22] but for purposes of our work we generalize it to decay speed.

The model in time-domain (2.22) is incomplete because it does ignore correlation between DMCs at different time-delays and is not valid for characterizing distribution of DMCs for real measurements. There exists bandwidth limitations of real system [22] which are neglected by (2.22).

Power spectrum density, i.e., the Fourier transform of (2.22) is introduced in [22] to overcome this limitation.

$$\Psi_{HH}(f_1, f_2) = \Psi_H(f_1 - f_2) = \Psi_H(\Delta f) = \frac{\beta}{B + i2\pi\Delta f} \cdot e^{-i2\pi\Delta f \frac{\tau_c}{L}} \tag{2.23}$$

L denotes number of complex bins in the channel impulse response and other parameters are very same as in 2.22.

Function (2.23) describes a spectral correlation between components of the channel transfer function having a distance $\Delta f = f_1 - f_2$ [22].

The covariance function of channel impulse response in time domain is

$$\psi_{hh}(t_1, t_2) = \psi_h(t_2)\delta(t_1 - t_2) \tag{2.24}$$

and after two Fourier transforms on (2.24) we get

$$\Psi_{HH}(f_1, f_2) = \psi_H(f_1 - f_2). \tag{2.25}$$

This should clarify the relationship of equations (2.23) and (2.22).

Later on in section about channel impulse simulator we utilize the covariance matrix $\mathbf{R} \in \mathbb{C}^{L \text{ x } L}$ [22]

$$\mathbf{R} = \begin{bmatrix} \Psi_H(0) & \Psi_H(-f_0) & \cdots & \Psi_H(-(L-1)f_0) \\ \Psi_H(f_0) & \Psi_H(0) & \ddots & \vdots \\ \vdots & \ddots & \ddots & \Psi_H(-f_0) \\ \Psi_H((L-1)f_0) & \cdots & \Psi_H(f_0) & \Psi_H(0) \end{bmatrix} \tag{2.26}$$

when generating signal featuring the dense multi-path components. This approach using frequency domain respects the limitations of measurements and generates realistic data featuring DMCs.

2.3.6 Complete channel model

Complete channel model consists of MPCs where each MPC has particular DMC associated to it and AWGN representing channel's background noise. As we have already mentioned, channel is characterized by its channel impulse response and in our work we refer to the channel model as the CIR. Algebraic equation describing CIR in continuous-time time domain is

$$h(t) = h_{mpc}(t) + h_{dmc}(t) + w(t) = \sum_{n=0}^{N} \left[h_{mpc}^{(n)}(t) + h_{dmc}^{(n)}(t) \right] + w(t) \tag{2.27}$$

where n iterates over the multi-path components, $h_{mpc}^{(n)}$ is the contribution of n$^{\text{th}}$ multi-path component, $h_{dmc}^{(n)}$ is the contribution of dense multi-path components associated to n$^{\text{th}}$ multi-path component and $\boldsymbol{w}$ is AWGN. The frequency domain form of (2.27) is

$$h[k] = h_{mpc}[k] + h_{dmc}[k] + w(t) = \sum_{n=0}^{N} \left[h_{mpc}^{(n)}[k] + h_{dmc}^{(n)}[k] \right] + w[k]. \tag{2.28}$$

Chapter 3

Cramér-Rao Lower Bound

When measuring performance of parameter estimation of a signal which features a random components it would be highly welcomed to have a tool that says how much information does a particular signal carries. By that we mean a theoretical limit of parameter estimation. Then we are able to compare our algorithms not only among them but also to the theoretical limit and state how close their estimates can get.

Such a theoretical limit broadly accepted and used [22],[8], [25] is the Cramér-Rao Lower Bound (CRLB) first introduced in [3] which expresses a lower bound on the variance of an unbiased estimator. Estimator is unbiased when

$$E[\hat{\boldsymbol{\xi}} - \boldsymbol{\xi}] = \mathbf{0} \tag{3.1}$$

where $\hat{\boldsymbol{\xi}}$ is vector of estimates and $\boldsymbol{\xi}$ are true value parameters.

CRLB states that the variance of an unbiased estimator is at least as high as the inverse of its Fisher information. Fisher information matrix consists of particular Fisher information entries which are values that describe how much information about certain parameter ξ is carried in a random variable X that depends upon ξ. Given a $f(X;\xi)$ as a probability density function for X conditioned on ξ.

Partial derivative with respect to ξ of log-likelihood function is called score.

$$\text{score} = E\left[\frac{\partial}{\partial \xi} \log_e(f(X;\xi)) \,\middle|\, \xi\right] \tag{3.2}$$

Variance of the score is then defined to be Fisher information.

$$\mathcal{I}(\xi) = \mathrm{Var}\left(E\left[\frac{\partial}{\partial \xi}\log_e(f(X;\xi)) \,\middle|\, \xi\right]\right) = E\left[\left(\frac{\partial}{\partial \xi}\log_e(f(X;\xi))\right)^2 \,\middle|\, \xi\right] \tag{3.3}$$

Formula computing CRLB of τ_n of n^{th} propagation path for our use-case is derived in [22] and is

$$\mathrm{CRLB}\tau_n = \frac{1}{\mathrm{SNR}_n}\frac{6}{(M^2-1)M} \tag{3.4}$$

where M is number of samples. CRLB bounds the variance of estimated parameter thus to compare it to RMSE of estimates we need to compute square-root and normalize it to the CIR length M to get the comparable dimension.

We introduce

$$\mathrm{CRLB}\tau_n{}^* = M\sqrt{\frac{1}{\mathrm{SNR}_n}\frac{6}{(M^2-1)M}} \tag{3.5}$$

for purposes of comparison using RMSE of an estimation.

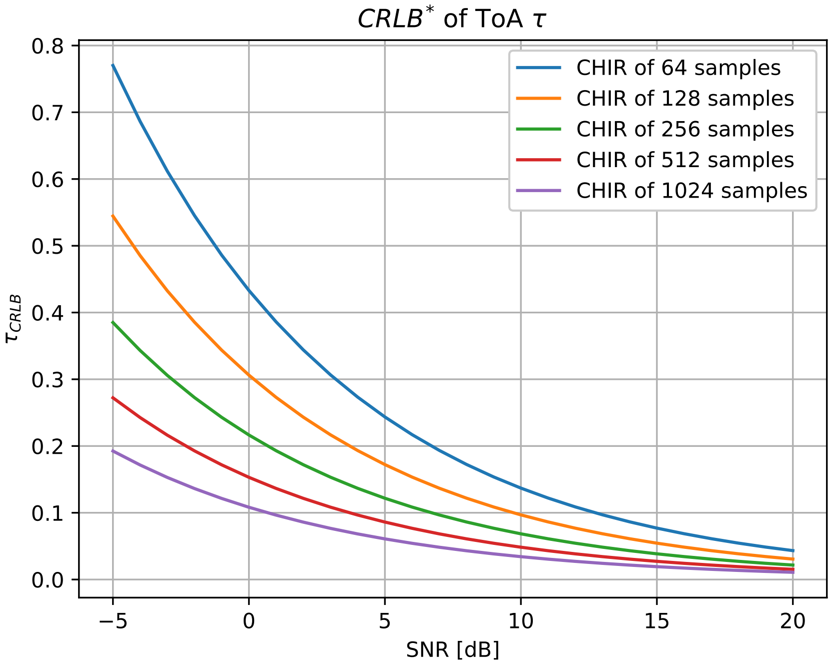

Figure 3.1: CRLB for various signal to noise ratios (SNRs) and various CIR length.

In fig. 3.1 one can see plotted values of theoretical estimation lower bound - CRLB$\tau_n{}^*$ for various signal to noise ratios (SNRs) and various length of CIRs.

That is a brief introduction which is not neither proper definition or rigorous derivation of either Fisher information and CRLB. Our research aims straightly at parameter estimation and we do not focus on theory background of CRLB and thus we only utilize CRLB as a tool and theory can be found in cited literature.

Part II

Practical part

Chapter 4

Channel impulse response simulator

In this chapter we introduce channel impulse response (CIR) simulator. It is very important that the simulator is implemented correctly and is based on correct algebraic equations. Both state-of-the-art method algorithm and our proposed method to estimate channel parameters is fed with simulator generated CIRs. We need to follow real world conditions and physical principles in order to obtain realistic channel impulse response but on the other hand we need to respect limitations of measurement equipment.

The simulator is implemented in Python (version 3.6) using Jet Brain's PyCharm Professional (version 2020.1.3) integrated development environment in Scientific mode. Among generally available Python packages we have used NumPy (version 1.19.2) for complex mathematical calculations and Matplotlib (version 3.3.2) visualisation. Structured and commented code is available as in thesis's attachment.

The tool is implemented in such a way that user provides a set of input parameters that either strictly define or bound the properties of dataset and generated samples of particular channel impulse response. Simulator generates and saves the dataset in binary file using NumPy *.npy. These generated datasets are then ready to easily use in further research in both state-of-the-art method's performance evaluation and neural network's training and testing because of its general format and included labels.

parameter	type	description
dataset_n	int	number of channel impulse response samples in dataset
sample_l	int	number of complex bins in one channel impulse response sample / length of one channel impulse response sample
mpc_n	int	number of multi-path components in one channel impulse response sample
mpc_snr_base	float	base value of SNR [dB] for multi-path components
mpc_snr_var	float	variance of SNR [dB] for multi-path components
dmc_trigger	boolean	dense-multi-path components on/off trigger
dmc_power_base	float	base value of relative power of dense-multipath components to associated multi-path component's amplitude
dmc_power_var	float	variance of relative power of dense-multipath components to associated multi-path component's amplitude
dmc_len_base	float	base value of length [number of complex-bins] of contribution/effect of dense-multipath components after associated multi-path component
dmc_len_var	float	variance of length [number of complex-bins] of contribution/effect of dense-multipath components after associated multi-path component
awgn_trigger	boolean	awgn on/off trigger

Table 4.1: List of channel impulse response simulator's input parameters.

The parameters whose names match *_base and *_var notation set boundaries for random variables that they describe. It is necessary to generate samples possessing random parameters and characteristics in order to simulate effects of various real world environments of propagation.

In following sections we will describe how are particular parameters used while generating the channel impulse response and how is the finite sample composed.

4.1 Generating multi-path components

Generating MPCs in time-domain using (2.8) would lead to limitation that we could only produce peaks at certain bins and not anywhere in between considering - discrete signal. That is not the case applicable for real measurement simulation, where the peaks are located no matter discrete sampling resolution.

Therefore we first generate N multi-path components in frequency domain using (2.13) where N equals to `mpc_n` in tab. 4.1 and

$$\alpha_n \sim \mathcal{U}(\alpha_n^{\text{MIN}}, \alpha_n^{\text{MAX}}) \tag{4.1}$$

$$\theta_n \sim \mathcal{U}(0, 2\pi) \tag{4.2}$$

$$\tau_n \sim \mathcal{U}(0, l) \tag{4.3}$$

where $\alpha_n^{\text{MIN}} =$ `mpc_snr_base`, $\alpha_n^{\text{MAX}} =$ `mpc_snr_base` $+$ `mpc_snr_var` and $l =$ `sample_l` are parameters from tab. 4.1. Function $\mathcal{U}$ denotes continues uniform distribution. Then using inverse Fourier transform we obtain time-domain based channel impulse response sample such as in fig. 4.2.

The amplitudes α_n in (4.1) are in absolute values while `mpc_snr_base` and `mpc_snr_var` are in decibels. Conversions is done using following formulas

$$\alpha_n = 10^{\frac{\alpha_n[\text{dB}]}{20}} \tag{4.4}$$

and

$$\alpha_n[\text{dB}] = 20 \log_{10}(\alpha_n). \tag{4.5}$$

Description of complete contribution of MPCs in (4.7) compatible notation is

$$h_{mpc}[k] = \mathcal{F}^{-1}\{H_{mpc}[k]\} = \mathcal{F}^{-1}\{\sum_{n=0}^{N} H_{mpc}^{(n)}[k]\} = \mathcal{F}^{-1}\{\sum_{n=0}^{N} \alpha_n e^{i\theta_n} e^{-i2\pi k\tau_n}\} \tag{4.6}$$

The complete flow diagram describing process of generating MPCs is in fig. 4.1.

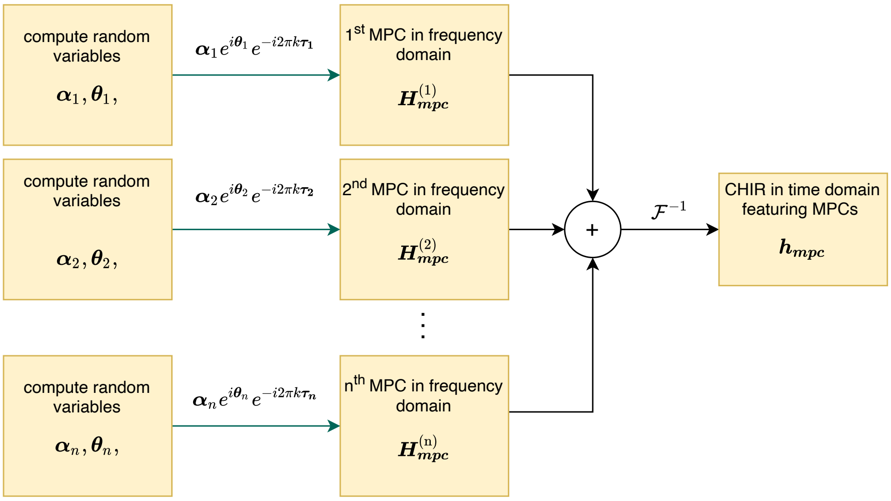
compute random variables
$\boldsymbol{\alpha}_1, \boldsymbol{\theta}_1,$
$\boldsymbol{\alpha}_1 e^{i\boldsymbol{\theta}_1} e^{-i2\pi k\boldsymbol{\tau}_1}$
1st MPC in frequency domain
$\boldsymbol{H}_{\boldsymbol{mpc}}^{(1)}$
compute random variables
$\boldsymbol{\alpha}_2, \boldsymbol{\theta}_2,$
$\boldsymbol{\alpha}_2 e^{i\boldsymbol{\theta}_2} e^{-i2\pi k\boldsymbol{\tau}_2}$
2nd MPC in frequency domain
$\boldsymbol{H}_{\boldsymbol{mpc}}^{(2)}$
+
$\mathcal{F}^{-1}$
CHIR in time domain featuring MPCs
$\boldsymbol{h}_{\boldsymbol{mpc}}$
compute random variables
$\boldsymbol{\alpha}_n, \boldsymbol{\theta}_n,$
$\boldsymbol{\alpha}_n e^{i\boldsymbol{\theta}_n} e^{-i2\pi k\boldsymbol{\tau}_n}$
nth MPC in frequency domain
$\boldsymbol{H}_{\boldsymbol{mpc}}^{(n)}$

Figure 4.1: Flow of generating MPCs.

By plugging (4.6) into (4.7) and constraining $h_{\mathrm{dmc}}[k] = 0$ and $w[k] = 0 \ \forall \ k$ we get a CIR sample that only consists of MPCs

$$h[k] = h_{mpc}[k] + h_{dmc}[k] + w[t] = h_{mpc}[k] = \mathcal{F}^{-1}\{\sum_{n=0}^{N} \alpha_n e^{i\theta_n} e^{-i2\pi k\tau_n}\}. \tag{4.7}$$

Such a sample can be observed in fig. 4.2 as a continuous plot created by interpolation in between discrete values for higher clarity.

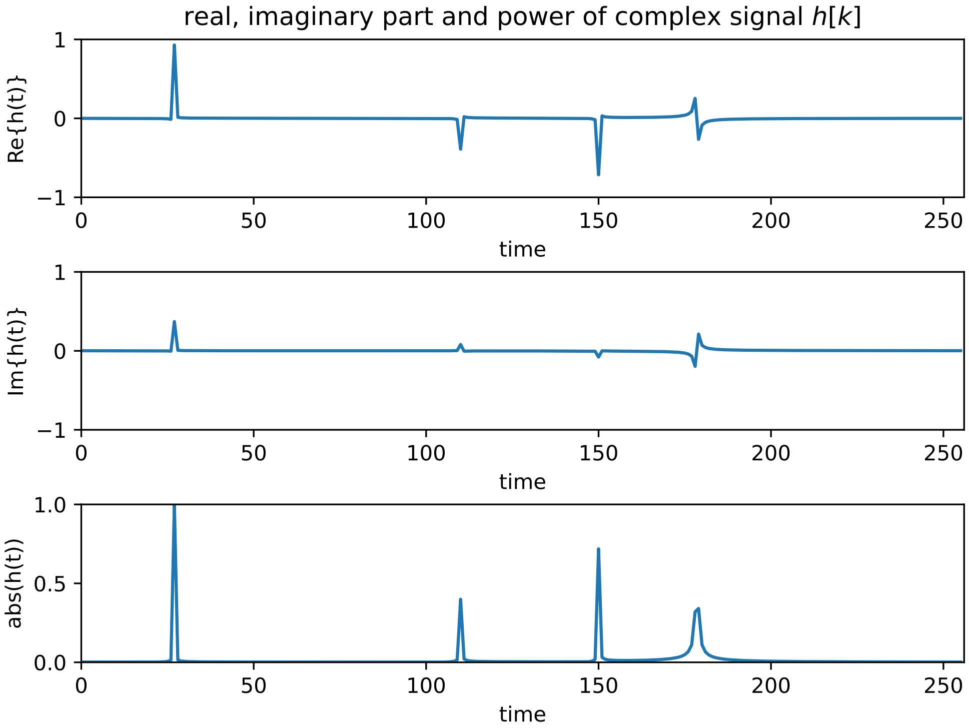

Figure 4.2: Example of generated channel impulse response in time-domain featuring only multi-path components.

4.2 Generating dense multi-path components

We have introduced two approaches both suitable for generating artificial realistic DMCs. The first (stochastic one) is based on observation of PDP of DMCs which has been proved to have exponential decay in time after the main peak, unfortunately this approach does not respect the limitation in measurement resolution and thus we make use of the second introduced approach that generates dense multi-path components as a noise with covariance matrix $\boldsymbol{R}$ (2.26).

Generating covariance matrix $\boldsymbol{R}$ for n$^{\text{th}}$ DMC associated to n$^{\text{th}}$ MPC consists of computing the matrix entries with (2.23) where the parameters β, B are specific to n$^{\text{th}}$ DMC and computed as

$$\begin{aligned}\boldsymbol{\beta}_n &\sim \mathcal{U}(\beta_n^{\mathrm{MIN}}, \beta_n^{\mathrm{MAX}})\\ \beta_n^{\mathrm{MIN}} &= \texttt{dmc\_power\_base}\\ \beta_n^{\mathrm{MAX}} &= \texttt{dmc\_power\_base} + \texttt{dmc\_power\_var}\end{aligned} \tag{4.8}$$

with parameters from tab. 4.1.

$$B_n = -\frac{\log_e(T)}{D_n} \tag{4.9}$$

We can not directly compute the decay speed of DMC B_n because it depends on T which specifies the final expected power of DMC that it reaches after L_n complex bins. We set $T = 0.01$ which means that all DMCs decay to one percent of their power after L_n complex bins which we consider to be such a low value that it does not affect the signal anymore because the contribution is completely lost in effects of channel random processes represented by AWGN. L_n are generated as a random values of Uniform distribution

$$\begin{aligned}D_n &\sim \mathcal{U}(L_n^{\mathrm{MIN}}, D_n^{\mathrm{MAX}})\\ D_n^{\mathrm{MIN}} &= \texttt{dmc\_len\_base}\\ D_n^{\mathrm{MAX}} &= \texttt{dmc\_len\_base} + \texttt{dmc\_len\_var}\end{aligned} \tag{4.10}$$

with parameters from tab. 4.1 and τ_n as ToA of n$^{\mathrm{th}}$ MPC.

By multiplying the covariance matrix $\boldsymbol{R}_n$ (2.26) of particular DMC by random vector $\boldsymbol{r}$ we get vector $\boldsymbol{H_{dmc}}^{(n)}$ that is essentially image of signal featuring DMC in frequency domain.

$$\boldsymbol{H_{dmc}}^{(n)} = \boldsymbol{r}_n^T \mathrm{chol}(\boldsymbol{R}_n) \tag{4.11}$$

where chol states for Cholesky decomposition.

The elements of random vector $\boldsymbol{r}$ are computed using Normal distribution with zero mean and variance α_n^2 so that it respects SNR of particular MPC.

$$\boldsymbol{r} \in \mathbb{R}^{L \times 1} \mid [r_1 ... r_n]^T \sim \mathcal{N}(0, \boldsymbol{\alpha}_n{}^2) \tag{4.12}$$

where $L = \texttt{sample\_l}$ from tab. 4.1.

Each CIR of particular n^{th} DMC is using inverse Fourier transformation (IFT) transformed into time domain. To obtain complete contribution of DMCs in (4.7) compatible notation, we sum the separate DMCs

$$\boldsymbol{h_{dmc}} = \sum_{n=0}^{N} \boldsymbol{h_{dmc}}^{(n)} = \sum_{n=0}^{N} \mathcal{F}^{-1}\{\boldsymbol{H_{dmc}}^{(n)}\} = \sum_{n=0}^{N} \mathcal{F}^{-1}\{\boldsymbol{r}_n^T \text{chol}(\boldsymbol{R}_n)\}. \tag{4.13}$$

Because IFT is linear, we make use of this fact

$$\boldsymbol{h_{dmc}} = \sum_{n=0}^{N} \mathcal{F}^{-1}\{\boldsymbol{r}_n^T \text{chol}(\boldsymbol{R}_n)\} = \mathcal{F}^{-1}\{\sum_{n=0}^{N} \boldsymbol{r}_n^T \text{chol}(\boldsymbol{R}_n)\} \tag{4.14}$$

and lower computational complexity so we only need to compute Fourier transformation (FT) once. This is how we compute DMCs but one could alway utilize previous formula (4.13) in need of obtaining each term describing particular DMC in time domain separately.

We have derived $\boldsymbol{h_{dmc}}$ as a complex-valued vector in (4.14). In order to utilize $h_{dmc}[k]$ notation such it is in (4.7) where we work with a discrete complex-valued function, we feel responsible for following clarification.

$$\boldsymbol{h_{dmc}} \in \mathbb{C}^{\text{ 1 x L}} \tag{4.15}$$

$$h_{dmc}[k] = [\boldsymbol{h_{dmc}}]_k = \left[\mathcal{F}^{-1}\{\sum_{n=0}^{N} \boldsymbol{r}_n^T \text{chol}(\boldsymbol{R}_n)\}\right]_k \tag{4.16}$$

The complete flow diagram describing process of generating DMCs is:

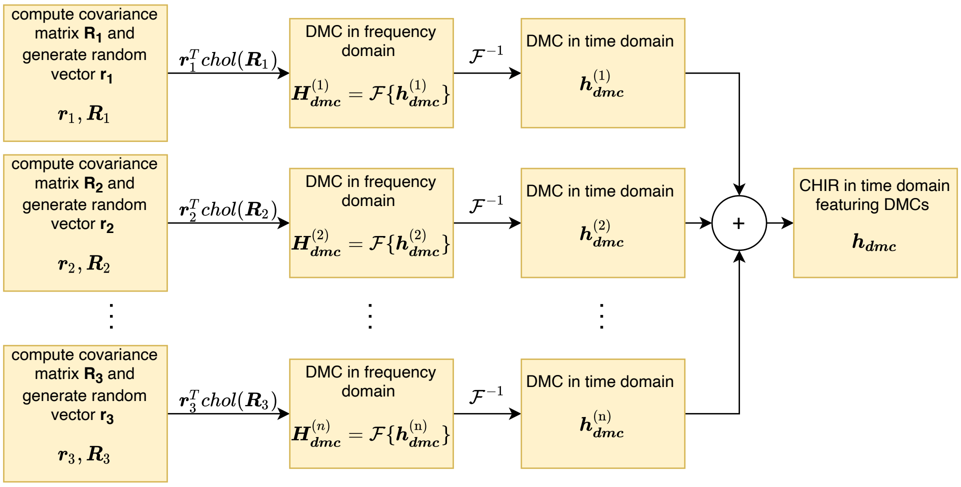
compute covariance matrix **R₁** and generate random vector **r₁**
$\boldsymbol{r}_1, \boldsymbol{R}_1$
$\boldsymbol{r}_1^T chol(\boldsymbol{R}_1)$
DMC in frequency domain
$\boldsymbol{H}^{(1)}_{dmc} = \mathcal{F}\{\boldsymbol{h}^{(1)}_{dmc}\}$
$\mathcal{F}^{-1}$
DMC in time domain
$\boldsymbol{h}^{(1)}_{dmc}$
compute covariance matrix **R₂** and generate random vector **r₂**
$\boldsymbol{r}_2, \boldsymbol{R}_2$
$\boldsymbol{r}_2^T chol(\boldsymbol{R}_2)$
DMC in frequency domain
$\boldsymbol{H}^{(2)}_{dmc} = \mathcal{F}\{\boldsymbol{h}^{(2)}_{dmc}\}$
$\mathcal{F}^{-1}$
DMC in time domain
$\boldsymbol{h}^{(2)}_{dmc}$
+
CHIR in time domain featuring DMCs
$\boldsymbol{h_{dmc}}$
compute covariance matrix **R₃** and generate random vector **r₃**
$\boldsymbol{r}_3, \boldsymbol{R}_3$
$\boldsymbol{r}_3^T chol(\boldsymbol{R}_3)$
DMC in frequency domain
$\boldsymbol{H}^{(n)}_{dmc} = \mathcal{F}\{\boldsymbol{h}^{(n)}_{dmc}\}$
$\mathcal{F}^{-1}$
DMC in time domain
$\boldsymbol{h}^{(n)}_{dmc}$

Figure 4.3: Flow of generating DMCs.

By plugging (4.16) into (4.7) and constraining $h_{\mathrm{mpc}}[k] = 0$ and $w[k] = 0$ we get

$$h[k] = h_{mpc}[k] + h_{dmc}[k] + w(t) = h_{dmc}[k] = \left[\mathcal{F}^{-1}\{\sum_{n=0}^{N} \boldsymbol{r}_n^T \mathrm{chol}(\boldsymbol{R}_n)\}\right]_k \tag{4.17}$$

which is a CIR sample that only consists of DMCs.

Such a sample can be observed in fig. 4.4 as a continuous plot created by interpolation in between discrete values for higher clarity.

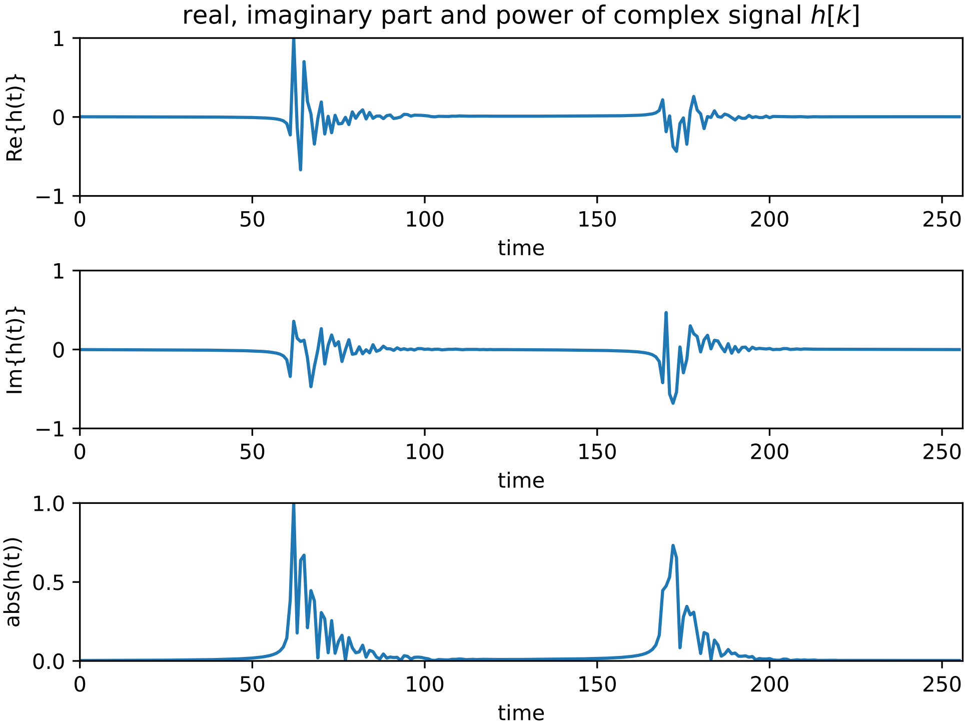

Figure 4.4: Example of generated CIR in time-domain featuring multi-path components and dense multi-path components.

4.3 Generating noise

We generate noise as a vector $\boldsymbol{w} \in \mathbb{C}$ of length $L =$ `sample_l` from tab. 4.1. To remain compatible with notation $w[k]$ used in (4.7) we need to define following transition. Relationship between complex-valued vector and complex discrete signal is as simple as

$$\boldsymbol{w}_k = w[k] \tag{4.18}$$

To obtain $\boldsymbol{w}$ we utilize formulas introduced in previous chapter (2.14) and (2.15).

$$\begin{aligned}\Re\{\boldsymbol{w}_k\} \sim \mathcal{N}(\mu, \sigma^2)\\ \Im\{\boldsymbol{w}_k\} \sim \mathcal{N}(\mu, \sigma^2)\end{aligned} \tag{4.19}$$

where we choose $\mu = 0$ and $\sigma^2 = \frac{1}{2}$ so the mean power of signal $\boldsymbol{w}$ equals to 1. This condition simplifies process of generating signal of certain SNR.

$$\boldsymbol{w}_k = \Re\{\boldsymbol{w}_k\} + j\Im\{\boldsymbol{w}_k\} \tag{4.20}$$

By plugging (4.20) into (4.7) and constraining $h_{\mathrm{mpc}}[k] = 0$ and $h_{\mathrm{dmc}}[k] = 0$ we get a CIR sample that only consists of AWGNs.

$$h[k] = h_{mpc}[k] + h_{dmc}[k] + w[k] = w[k] \tag{4.21}$$

Such a sample can be observed in fig. 4.5 as a continuous plot created by interpolation in between discrete values for higher clarity.

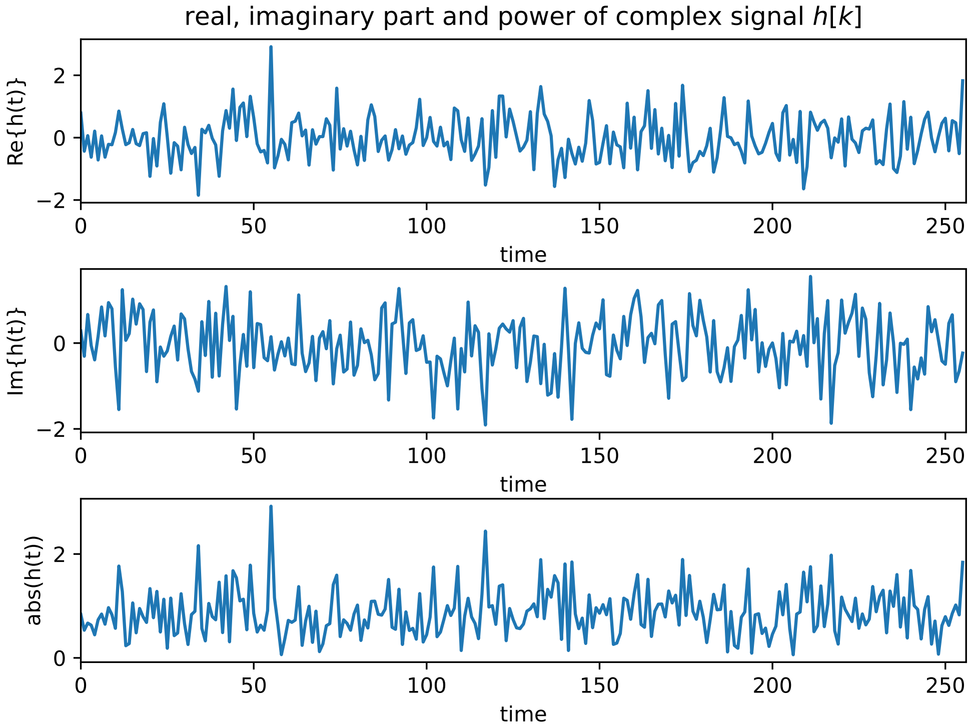

Figure 4.5: Example of generated channel impulse response in time-domain featuring only additive white Gaussian noise.

4.4 Complete channel

Generating complete CIR featuring all MPCs, DMCs and AWGN is based on complete CIR algebraic model for discrete signal described in (4.6) where the particular terms are generated according to (4.6), (4.16) and (4.21) respectively.

When we put these equations in the (4.7) we obtain:

$$h[k] = \mathcal{F}^{-1}\{\sum_{n=0}^{N} \alpha_n e^{i\theta_n} e^{-i2\pi k\tau_n}\} + \left[\mathcal{F}^{-1}\{\sum_{n=0}^{N} \boldsymbol{r}_n^T \mathrm{chol}(\boldsymbol{R}_n)\}\right]_k + w[k] \quad (4.22)$$

.

Such a sample can be observed in fig. 4.6 as a continuous plot created by interpolation in between discrete values for higher clarity.

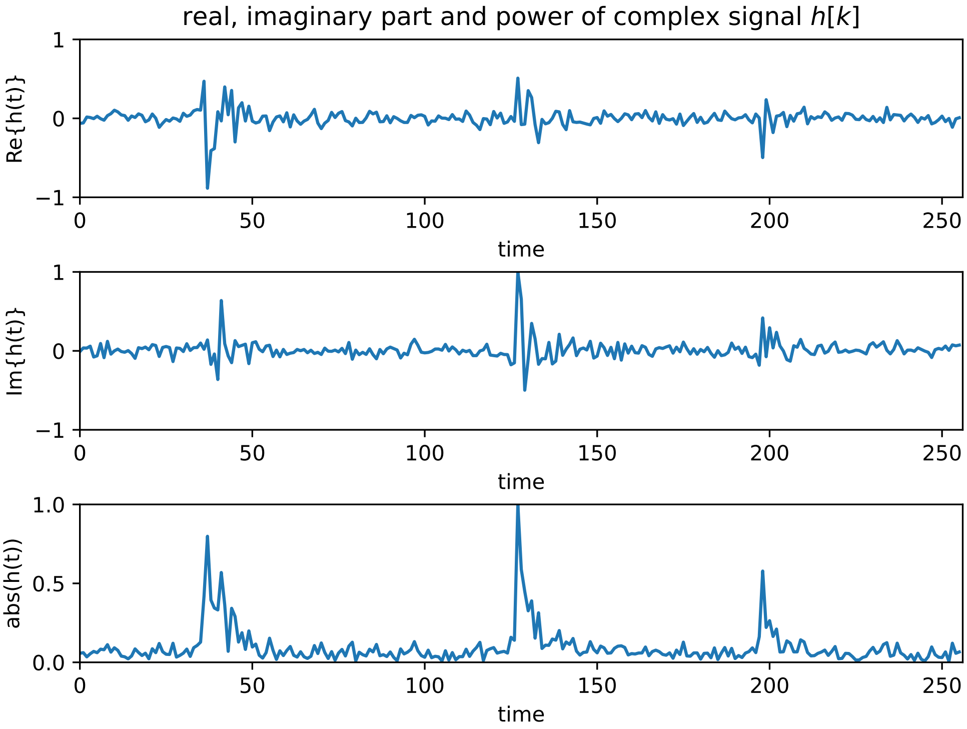

Figure 4.6: Example of generated channel impulse response in time-domain featuring multi-path components and dense multi-path components and additive white Gaussian noise.

We have mentioned that while generating AWGN we choose its parameters so the mean power of the generated AWGN signal equals to 1. Nevertheless one can see that in fig. 4.6 the mean power of its AWGN component is equal to by far less than 1. After the final signal is composed we normalize its real and imaginary part in order to fit the values into the interval:

$$\begin{aligned} -1 \leq &\Re\{h[k]\} \leq 1 \; \forall \; k \\ -1 \leq &\Im\{h[k]\} \leq 1 \; \forall \; k \end{aligned} \quad (4.23)$$

We do not loose any information or lower resolution while normalizing the signal. There is no downside in doing so when using conventional algorithms to estimate channel parameters, on the other hand there is big advantage to have signal normalized when feeding it as an input to a neural network.

4.5 Labels

Apart of only generating CIRs we also need information containing the correct values of parameters such are for example ToAs in order to be able to measure performanace of our estimation methods and train ANNs.

Exact forms of labels will be introduced when used in chapter about parameter estimation using ANNs.

Chapter 5

Channel parameter estimation

Complete generated CIR signal (4.22) features parameters τ_n, α_n, θ and N as number of MPCs, DMCs. The focus of our work is on estimating parameters ToAs $\boldsymbol{\tau}$ and parameters of DMCs. Amplitudes denoted by $\boldsymbol{\alpha}$ and phase-shifts $\boldsymbol{\theta}$ are linear parameters and can be obtained by BLUE according to [22], [14]. In our case both $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$ are uniformly distributed and not correlated. They don't carry any information about the channel but for purposes of estimating $\boldsymbol{\tau}$ we need to remember and estimate them.

We introduce and describe SAGE algorithm which is proved [6] to follow CRLB (3.4) when there is appropriate spacing between peeks (MPCs) [3]. We need to know the number of MPCs apriory and provide initial estimates for SAGE. Two problems appear when using SAGE algorithm: first is that in real world usage we wouldn't know number of peaks apriory and second is that iterations of SAGE algorithm estimating multiple $\boldsymbol{\tau}$ could converge to the same peak [22]. This is a considerable downside of SAGE algorithm approach. Later on we introduce our approach based on ANNs where we first roughly estimate ToAs of MPCs and power delay profile of DMCs. Using these results as augmented dataset together with original CIR we fine estimate ToAs $\boldsymbol{\tau}$. Fine estimation is not done on complete sample of CIR but on a cropped CIR - window surrounding the particular MPC. We thus bypass the problem of knowning number of components apriory. When it comes to real-time processing it is significant advantage of ANNs that they produce result in constant time and are thus usable in such applications.

Chapter 6

State-of-the-art method for channel parameter estimation

In this chapter we introduce the SAGE algorithm which will serve us as a performance reference algorithm when processing results of our ANNs parameter estimation method. SAGE algorithms is proved to follow CRLB so it is indeed the best reference estimator in matter of results. SAGE is an iterative algorithm with high time complexity which comes as a downside if considering real-time processing.

6.1 EM algorithm

The core of SAGE [6] algorithm lies in expectation–maximization (EM) algorithm [4] so it is necessary to introduce the later first. EM is an iterative method solving maximum likelihood (ML) estimation problems where part of the information is missing, unknown or noised. EM estimates complete unobserved data from incomplete observed data and consists of expectation and maximization step.

There is discussion if the EM algorithm is an actual algorithm since it does only provide a general approach and not particular solution.

The basic concept the EM algorithm relies on is that we have complete but unobserved data and incomplete but observed data. Imagine you measured a

CIR - that is the observed part and it is incomplete because the sample is corrupted by finite resolution of measurement equipment, noised by various random processes and therefore does not contain the complete information. On the other hand we know the channel model perfectly (4.22) (channel model is also an approximation but we consider it to be the true reference for our signal) but can not measure such a CIR - that is the complete but unobserved part.

Proper derivation of EM estimating channel parameters can be found in [7].

6.2 SAGE algorithm

SAGE states for space-alternating generalized expectation-maximization. The algorithm was first introduced in [6]. Each iteration of SAGE algorithm is essentially a iteration no EM algorithm. SAGE comes with with great complexity relief with rising number of estimating parameters of $\boldsymbol{\xi}_i$ because it decomposes the optimization problem into several less complex sub-problems. When performing the maximization step of EM algorithm the likelihood function might be very hard to optimize because of high dimensionality of likelihood function which is given by the number of model parameters.

SAGE is given a rough estimation of $\hat{\boldsymbol{\xi}}^{(0)}$ for the zeroth iteration and its performance is strongly determined on how sufficiently good these initial estimates are [6]. This implies that we need to know the number of signal components - MPCs which is not part of our research so we work with apriory known value. Because we only estimate ToAs the vector $\boldsymbol{\xi}$ is essentialy a vector of particular time delays $\boldsymbol{\tau}_0 \dots \boldsymbol{\tau}_n$ with a length of N which is number of MPCs in CIR.

Algorithm 1: SAGE algorithm

```
input  : $\hat{\boldsymbol{\xi}}^{(0)}\Big|_{\mu=0}$ - initial guess of parameters, $\boldsymbol{y}$ - CIR
output : $\hat{\boldsymbol{\xi}}^{(\mu)}\Big|_{\mu=\mu_{\mathrm{END}}}$ - final estimation of parameters
$\mu = 0$;
while not converged do
    compute $\boldsymbol{B}$ based on $\hat{\boldsymbol{\xi}}^{(\mu)}$ (6.1);
    for $i$ as each $n^{th}$ signal component do
        $\boldsymbol{Y} = \boldsymbol{y}$;
        for $j$ as each $n^{th}$ signal component do
            if $i \neq j$ then
                $\boldsymbol{Y_j} = \boldsymbol{Y_j} - \boldsymbol{B_j}$
            end
        end
        compute maximum likelihood estimate (MLE) of $\hat{\boldsymbol{\xi}}_n^{(\mu+1)}$ as 1-D
          minimization (6.4);
    end
    $\mu = \mu + 1$;
end
```

Pseudocode in alg. 1 describes SAGE. It takes CIR and initial guess of parameters $\hat{\boldsymbol{\xi}}^{(\mu)}$ where μ denotes iteration as an input. Then matrix $\boldsymbol{B}$ is computed

$$\boldsymbol{B} \in \Re^{\text{ N x N}} \mid \boldsymbol{B}_n(\tau_n) = \boldsymbol{h} \tag{6.1}$$

where $\boldsymbol{B}$ consists of n vectors $\boldsymbol{B}_n$ that are functions of τ_n representing n separate expected CIRs $\boldsymbol{h}$ (2.12) (each containing only n^{th} MPC and associted DMC) constructed using parameters of actual iteration $\hat{\boldsymbol{\xi}}^{(\mu)}$ with one vector being replaces with a CIR of τ as the parameter of minimizing function. Then parameters are optimized one by one. First $\boldsymbol{Y}$ is computed in a way that we basically from given input $\boldsymbol{y}$ which we subtract from all particular expected CIRs in B but the one we are optimizing. If our estimates $\hat{\boldsymbol{\xi}}^{(\mu)}$ are sufficiently good, we should be left with a single MPC in $\boldsymbol{Y}$ upon which we obtain MLE of $\hat{\boldsymbol{\xi}}^{(\mu+1)}$ as

$$\hat{\boldsymbol{\xi}}^{(\mu+1)} = \operatorname{argmin}_{\hat{\boldsymbol{\xi}}} \left(\boldsymbol{Y}^H\boldsymbol{R}^{-1}\boldsymbol{X} - \left(\boldsymbol{Y}^H\boldsymbol{R}^{-1}\boldsymbol{B}\right)\left(\boldsymbol{B}^H\boldsymbol{R}^{-1}\boldsymbol{B}\right)^{-1}\left(\boldsymbol{B}^H\boldsymbol{R}^{-1}\boldsymbol{Y}\right)\right) \tag{6.2}$$

according to [22] where $\boldsymbol{R}^{-1}$ is inverse of covariance matrix of $\boldsymbol{Y}$, which fulfils $\boldsymbol{R} = \sigma^2\boldsymbol{I}$ because signal entries are not correlated $\boldsymbol{R}$ is diagonal. SAGE algorithm ignores DMCs and pure AWGN of power equal to 1 is expected to be in background and thus $\sigma = 1 \implies \boldsymbol{R} = \boldsymbol{I}$. Therefore we state that

$$\hat{\boldsymbol{\xi}}^{(\mu+1)} = \operatorname{argmin}_{\hat{\boldsymbol{\xi}}} \left(\boldsymbol{Y}^H\boldsymbol{Y} - \left(\left(\boldsymbol{Y}^H\boldsymbol{B}\right)\left(\boldsymbol{B}^H\boldsymbol{B}\right)^{-1}\left(\boldsymbol{B}^H\boldsymbol{Y}\right)\right)\right) \tag{6.3}$$

where $\hat{\xi}_n^{(\mu+1)} = \hat{\tau}_n^{(\mu+1)}$ and

$$\hat{\tau}_n^{(\mu+1)} = \operatorname{argmin}_{\hat{\tau}} \left(\boldsymbol{Y}^H\boldsymbol{Y} - \left(\left(\boldsymbol{Y}^H\boldsymbol{B}(\hat{\tau}_n)\right)\left(\boldsymbol{B}(\hat{\tau}_n)^H\boldsymbol{B}(\hat{\tau}_n)\right)^{-1}\left(\boldsymbol{B}(\hat{\tau}_n)^H\boldsymbol{Y}\right)\right)\right) \tag{6.4}$$

6.2.1 Performance

We have generated 52 separate dataset with parameters specified in tab. 6.1 where 26 of them are datasets with SNR $\in \{-5, -4, \ldots 19, 20\}$[dB] for both cases: either containing and not containing DMCs.

parameter	value
`dataset_n`	1000
`sample_l`	256
`mpc_n`	1
`mpc_snr_base`	{-5, -4 ... 19, 20}
`mpc_snr_var`	0
`dmc_trigger`	{False, True}
`dmc_power_base`	0.25
`dmc_power_var`	0.5
`dmc_len_base`	16
`dmc_len_var`	32
`awgn_trigger`	True

Table 6.1: List of channel impulse response simulator input parameters used to generate datasets for SAGE performance evaluation.

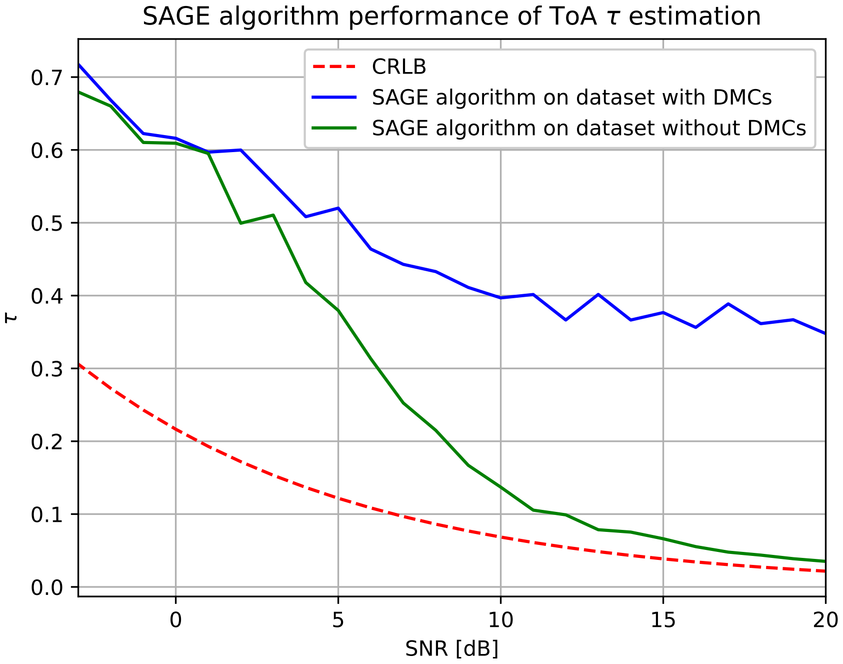

Figure 6.1: Sage performance on dataset of various SNRs when containing and not not containing DMCs.

Results of comparison can bee seen in fig. 6.1 where one could see that the performance on dataset without DMCs is asymptotically converging to CRLB in space of higher SNRs but there is a estimation margin on dataset with DMCs. If we knew the covariance matrix of input signal we could utilize the information in (6.2) but unfortunately we do not know the information apriori. Around SNR[dB] equal to 0 (where the signal is expected to have same power as noise) both curves meet. Our interpretation is following: when the noise power level is about the same power of actual MPCs it does not really matter if there are any additional DMCs present. DMCs are modelled as an increased noise and are totally lost in the AWGN.

Chapter 7

Channel parameters estimation using artificial neural networks

7.1 Introduction to ANNs

Let us briefly summarize history of ANNs [26]. It all begun in 1943 Warren McCulloch a neurophysiologist and Walter Pitts a young mathematics described a mathematical model of how neuron could work [17]. Very early stage of research and developement work in this field has been done by Donald Hebb, Nathanial Rochester and John von Neumann until the year 1958 when the simplest neural network still used today called perceptron was introduced by Frank Rosenblatt. A year later in 1959 researchers Bernard Widrow and Marcian Hoff based in Sanford came up with two models ADALINE and MADALINE - the very first neural networks applied on real world's problems. Then research somewhat halted, it didn't bring up any significant progress because of low complexity and performance of electronics. Computing power and electronic's compelxity got huge push forward in following years and during these times the research was mostly done theoretically and practical research could be done on only very simple ANNs. Conferences and meetings about ANNs started happening across the whole world from Japan to USA as soon as in 1980s and researchers invented modern architerctures such as convolution neural network (CNN) [9] by Fukushima, Kunihiko, Long Short-Term Memory (LSTM) [10] by Sepp Hochreiter and Jürgen Schmidhuber and Gradient based learning by Yann LeCun [16].

Neural network is at its simplest and most general form is a function mapping space of inputs to the space of outputs. Where in our case space of inputs might be CIRs and space of outputs might be the channel parameter estimates $\boldsymbol{\xi}$.

$$f(\boldsymbol{\psi}) : \boldsymbol{X} \rightarrow \boldsymbol{Y} \tag{7.1}$$

where $\boldsymbol{X}$ is domain of inputs and $\boldsymbol{Y}$ is domain of outputs and f is a general function. Because f is defined as general we do not know neither its structure or its parameters $\boldsymbol{\psi}$.

For instance one could consider linear function

$$y = f(x) = ax + b \tag{7.2}$$

as a very simple form of neural network where we have a and b as ANN's parameters. We would then aim to find sufficient set of a, b which makes f work well for our application - we would perform a linear regression based on given $\boldsymbol{x}, \boldsymbol{y}$.

Both, choosing the right structure of f which is commonly referred to as ANN's architecture and finding suitable set of function parameters $\boldsymbol{\psi}$ which is commonly referred to as ANN's training is crutial for any application.

7.1.1 Architecture

Layers

Today there is a broadly accepted concept of describing sequential ANN's architecture by its layers:

$$f : \boldsymbol{X}_0 \xrightarrow{\zeta_1(f_1)} \boldsymbol{X}_1 \xrightarrow{\zeta_2(f_2)} \boldsymbol{X}_2 \xrightarrow{\zeta_3(f_3)} \dots \xrightarrow{\zeta_k(f_K)} \boldsymbol{X}_{\boldsymbol{K}} = \boldsymbol{Y} \tag{7.3}$$

where f are particular function in a layer of defined structure but not bound in parameters in between them and $\boldsymbol{X}$ might be arbitrary tensors. The reason why we need layers is the fact that sequential ANNs are often a combination of mostly linear layer functions $\boldsymbol{f}$ and non-linear activation functions $\boldsymbol{\zeta}$ which together create hugely complex non-linear functions that fit to particular applications. If we stuck linear layers one by one right after the previous linear layer we wouldn't make the ANN any more complex because one can substitute any combination of linear layers by one linear layer.

This (7.3) sequence of non-linear functions applied on the input creates a complex non-linear function that with right parameters $\boldsymbol{\psi}$ of $\boldsymbol{f}$ can be used to solve a particular task such is image recognition, text classification and much more. In our case it is a channel parameter estimation.

Apart of sequential models of ANN's there exists others such are for example recurrent neural network (RNN) introduced first in 1982 by John Hopfield in [11].

In following text we will shortly introduce concept of dense layer and 1-D convolution layer which we use for parameter estimation.

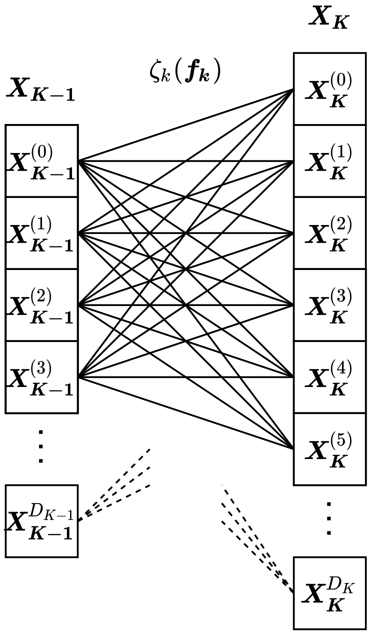

Figure 7.1: Dense layer of ANN.

$$\boldsymbol{y} = \zeta\left(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}\right) \tag{7.4}$$

Dense layer in fig.7.1 (also called fully connected layer) is a simple but most complex linear layer where one particular element of a vector (matrix) $\boldsymbol{X}_K$ is computed as a sum of a affine functions $f(x) = ax + b$ of independent parameters a,b for each connection where x are all elements from a vector (matrix) $\boldsymbol{X}_{K-1}$. The output of each affine function if fed as an input to a non-linear activation function ζ (7.4). Dense layer is very general form of ANN's layer and thus does not have any special applications.

Convolution layer in fig. 7.2 might be of arbitrary dimension (figure contains 2-D convolution) and is based on its kernel (blue area) which move all over the input vector (matrix) $\boldsymbol{X}$ and computes output as an element of vector (matrix) $\boldsymbol{X}_{K+1}$. In the example figure the element $\boldsymbol{X}_{K+1}^{(1,1)}$ is computed as.

$$\boldsymbol{X}_K = \zeta(c_{1,1}\boldsymbol{X}_K^{(1,1)} + c_{1,2}\boldsymbol{X}_K^{(1,2)} + c_{2,1}\boldsymbol{X}_K^{(2,1)} + c_{2,2}\boldsymbol{X}_K^{(2,2)}) \tag{7.5}$$

where ζ is non-linear activation function and ci, j are kernel parameters which are fixed for whole input. It is possible to create multiple kernels that are applied to the input one by one separately and create new dimensions (channels) of the data.

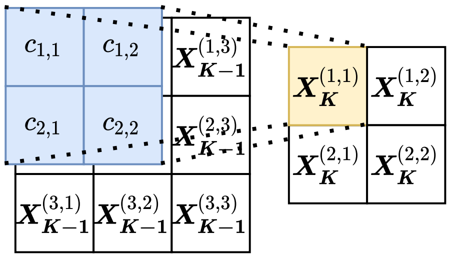
$c_{1,1}$
$c_{1,2}$
$c_{2,1}$
$c_{2,2}$
$\boldsymbol{X}_{K-1}^{(1,3)}$
$\boldsymbol{X}_{K-1}^{(2,3)}$
$\boldsymbol{X}_{K-1}^{(3,1)}$
$\boldsymbol{X}_{K-1}^{(3,2)}$
$\boldsymbol{X}_{K-1}^{(3,3)}$
$\boldsymbol{X}_{K}^{(1,1)}$
$\boldsymbol{X}_{K}^{(1,2)}$
$\boldsymbol{X}_{K}^{(2,1)}$
$\boldsymbol{X}_{K}^{(2,2)}$

Figure 7.2: Convolution layer of ANN.

The main application of CNNs nowadays is in image processing where CNNs are performing well in pattern recognition.

Activation functions

Activation functions are important part in between mostly linear layers. They allow us to create very complex networks by combining big amounts of non-linear functions in chains. Their structure is appropriate so they can serve as activators of certain connections. There exist a lot of types of activation functions which some serve better in a particular subspace of ANNs applications and some worse. Two types that are frequently used these days and we use in our research are rectified linear unit (ReLU), leaky rectified linear unit (Leaky ReLU) and Sigmoid activation functions fig. 7.3.

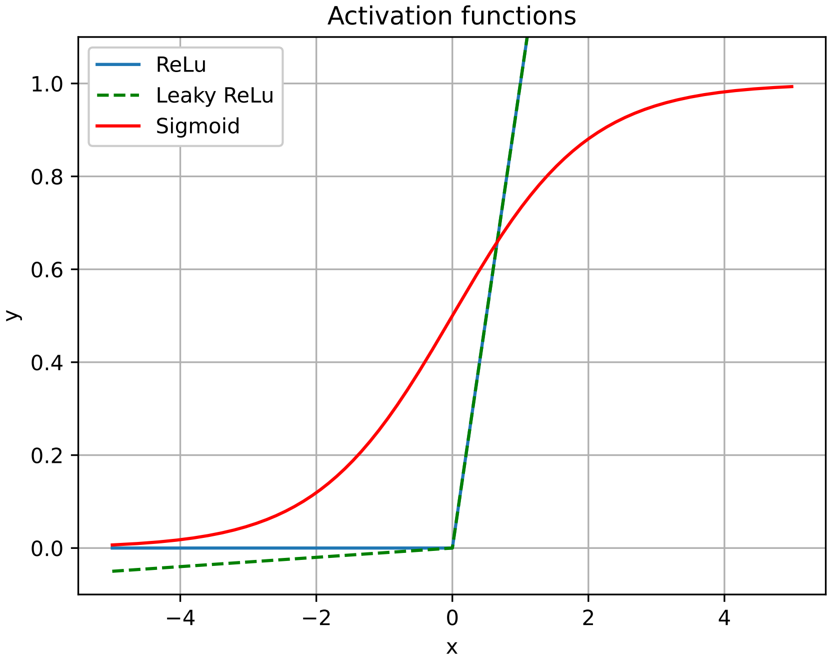

Figure 7.3: Activation functions.

Algebraic forms of introduced activation functions are following:

$$S(x) = \frac{e^x}{e^x + 1} \tag{7.6}$$

for Sigmoid activation function,

$$R(x) = \max(0, x) \tag{7.7}$$

for ReLU activation function and

$$LR(x) = \begin{cases} ax & x < 0 \\ x & x \geq x \end{cases} \tag{7.8}$$

for Leaky ReLU activation function where a is a parameter of a small value e.g. $a = 0.001$ creating a very gentle negative slope which prevents gradients to zero-out during network training which is a downside of ReLU activation function.

7.2 ANN training

We refer to training of an ANN as process of finding suitable set of parameters of certain ANN $\boldsymbol{\psi}$ (7.1). Suitable set of parameters is a set which makes the

ANN output expected values for given input. Finding $\boldsymbol{\psi}$ is a process when we using given training dataset, which consists of ANN's input and associated desired outputs so called labels, extract information of how particular elements of the $\boldsymbol{\psi}$ affect the ANN's output and tune them in order to get obtain better estimation.

7.3 ANN roughly estimating ToA of MPCs

In this section we introduce ANN which rougly estimates ToAs of MPCs. By roughly we mean estimation within limits of CIR sample resolution. ANN takes real, imaginary component and computed signal's power (2.20) as inputs. These are vectors of a certain length and output is also a vector of the very same length where its values indicate presence of peaks. Example in fig. 7.4. One can observe that true peaks that can be easily identified in signal power are detected by the ANN and its output marks the peak location. Very tiny noise in ANN's output can be seen around values 55 and 140.

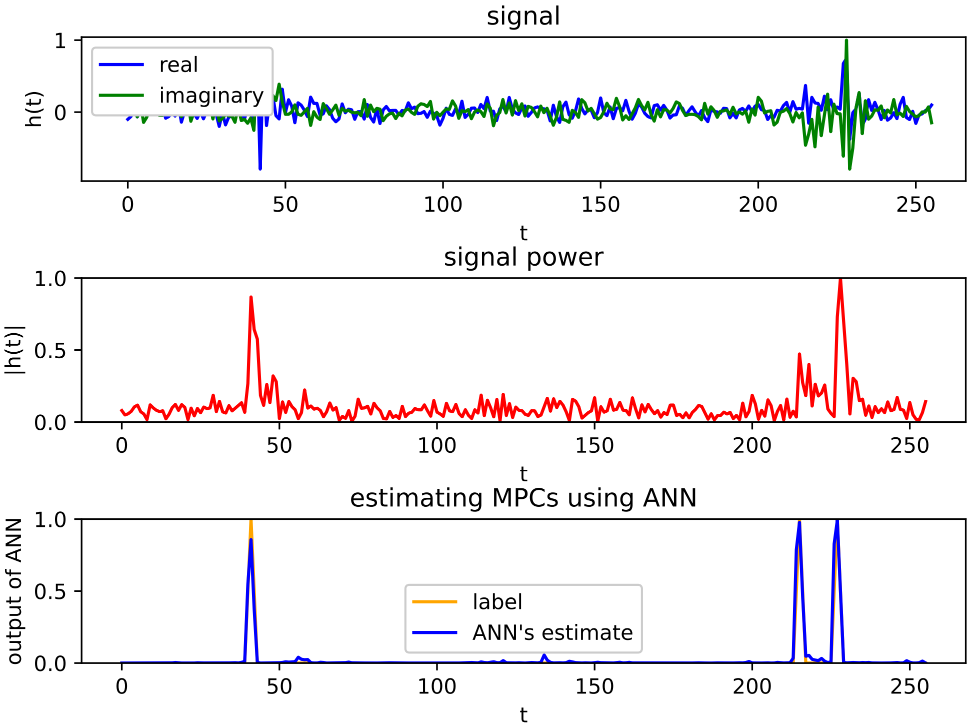

Figure 7.4: Input-output example of ANN rougly estimating ToAs of MPCs. The t is a number of ANN's input-output which corresponds to time delay normalized to sampling period.

We can not say that the ouput of the ANN is probability of peak presence because in case of multiple peaks the values would have to spread among several peaks in order to remain probability property of summing to 1 but considering one element of the output vector one could interpret its value as a probability.

7.3.1 Architecture

Inspired by pattern recognition oriented ANNs, we constructed our ANN as a sequential model containing several layers of 1-D convolutions together with Leaky ReLUs activation functions followed by one dense layer with Sigmoid activation function at its output.

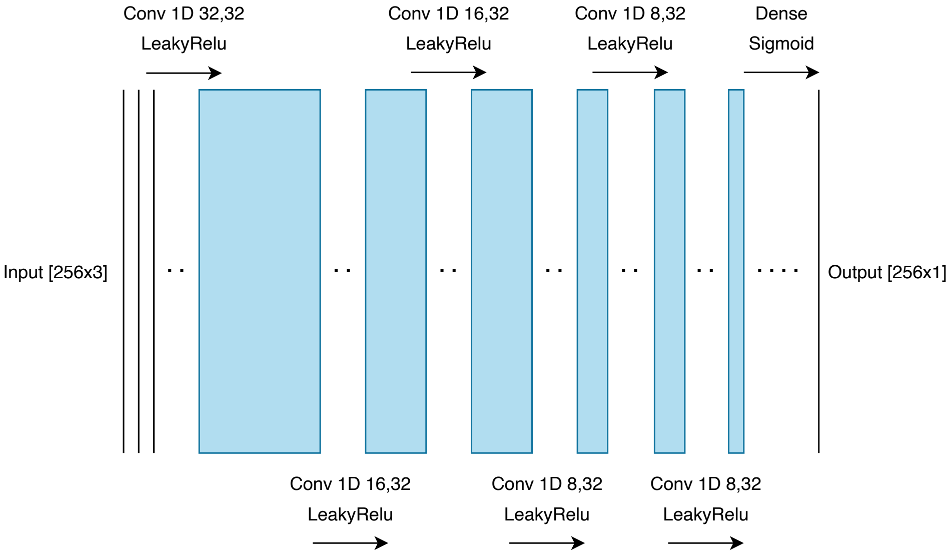

Figure 7.5: Architecture of ANN rougly estimating ToAs of MPCs.

7.3.2 Train and test dataset

We have generated train and test datasets specified in tab. 7.1 and trained the network using Adam optimizer [15] which has been proven as a well working complex optimizing method for training of broad field of ANNs and binary cross-entropy as loss function.

parameter	value
`dataset_n`	100000 for train and 10000 for test
`sample_l`	256
`mpc_n`	$\{1\ldots3\}$
`mpc_snr_base`	0
`mpc_snr_var`	20
`dmc_trigger`	True
`dmc_power_base`	0.25
`dmc_power_var`	0.5
`dmc_len_base`	16
`dmc_len_var`	32
`awgn_trigger`	True

Table 7.1: List of channel impulse response simulator's input parameters used to train ANN roughly estimating ToAs of MPCs.

7.3.3 Performance

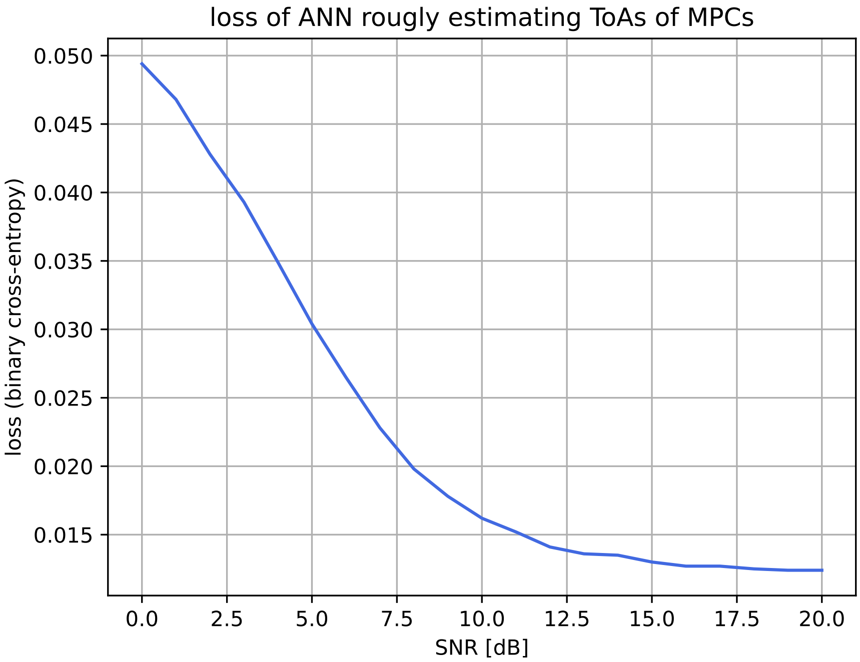

Figure 7.6: Loss of ANN rougly estimating ToAs of MPCs.

Loss of the ANN showed in fig 7.6 does not say much information apart of the fact that performance increase is significant in between $0 \leq$ SNR[dB] ≤ 7.5, then the performance gain is slowing down and by the time when SNR[dB] $=$ 20 the curve is almost flat. This is behavior is expected - the less noisy CIR as the input the better estimation.

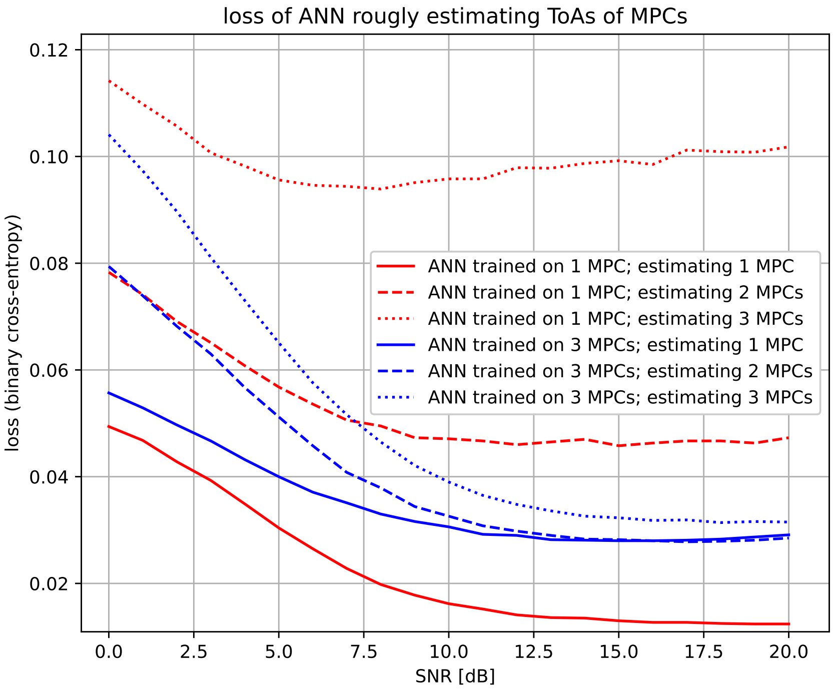

Figure 7.7: Loss of ANN rougly estimating ToAs of MPCs.

To demonstrate performance of the introduced ANN when multiple MPC are present we have trained the ANN using several datasets (still based on parameters in tab. 7.1) which each possess static (inside of particular dataset) number of MPCs but goes from 0 to 5. In fig. 7.7 we present two trained ANNs, first trained on dataset with 1 MPC (red) and second trained on dataset with 3 MPCs (blue). Performance of these two ANNs shows that to obtain best results one should train the network on a maximum expected number of MPCs because it provides decent estimations even if the CIR has less MPCs than the number it was trained on. On the other hand ANN's loss increases dramatically when estimating higher number of MPCs than the number it was trained on.

The output of this ANN is mainly used as an input to ANN fine estimating ToAs of MPCs introduced in later section but a standalone application could be providing initial estimates of ToAs of MPCs for example to the SAGE algorithm introduced earlier. ANN is able to tell apart peaks which are pure noise and peaks produced by MPCs even if the power of noise-peak is higher than the power of MPC-peak, such a sample can be seen in fig. 7.8.

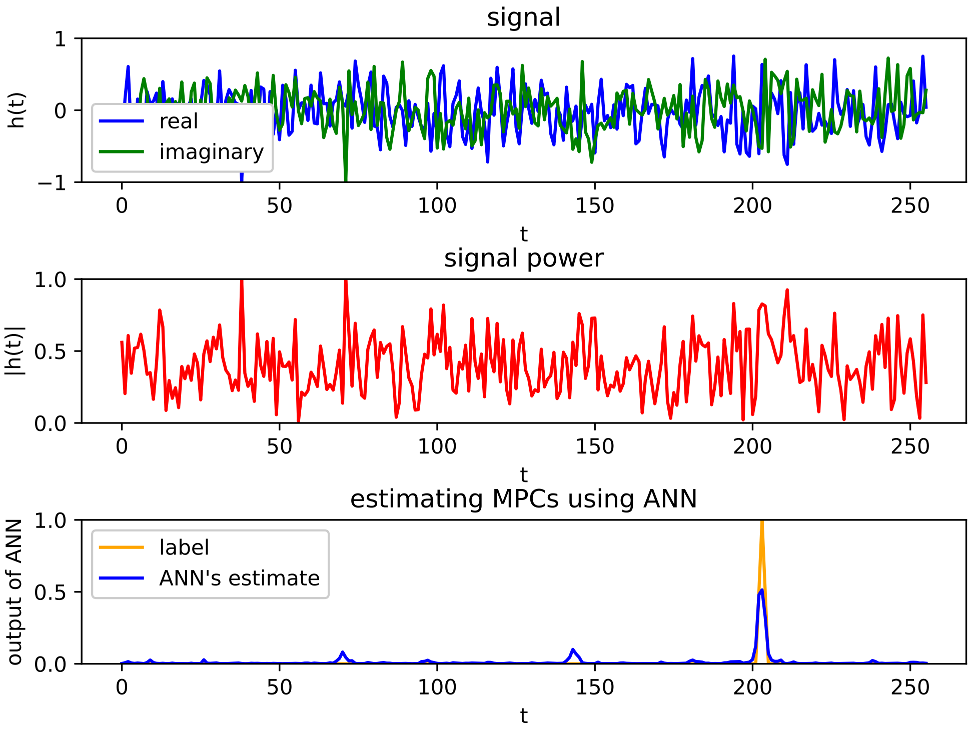

Figure 7.8: ANN estimating ToA of MPC in high power noise. The t is a number of ANN's input-output which corresponds to time delay normalized to sampling period.

It is not possilbe to esimate MPC's location just by looking at the real, imaginary part and signal's noise in fig 7.8 because the peak is completely lost in surrounding AWGN which produces even higher peaks. Our proposed ANN was however able to recognise the hidden MPC and mark it at its output. Label is going up to value 1.0 when the ANN's estimate is only half of it but still it is the highest element of ANN's output vector and thus usable for estimation. Apart of this, there are several marked positions where a MPC could according to ANN be, such position are rougly around 65 and 140.

7.4 ANN roughly estimating DMCs

In this section we introduce ANN which estimates power delay profile of dense multi-path components. The ANN takes real and imaginary components as inputs together with computed signal power (2.20). These are vectors of a certain length and outputs vector of very same length where values indicate presence of peaks. Input-output example of the ANN in fig. 7.9. One can see that our ANN is capable of locating DMCs and marking their PDP. If the SNR is high enough such it is here, apart of location of DMC the values of ANN output remain zero.

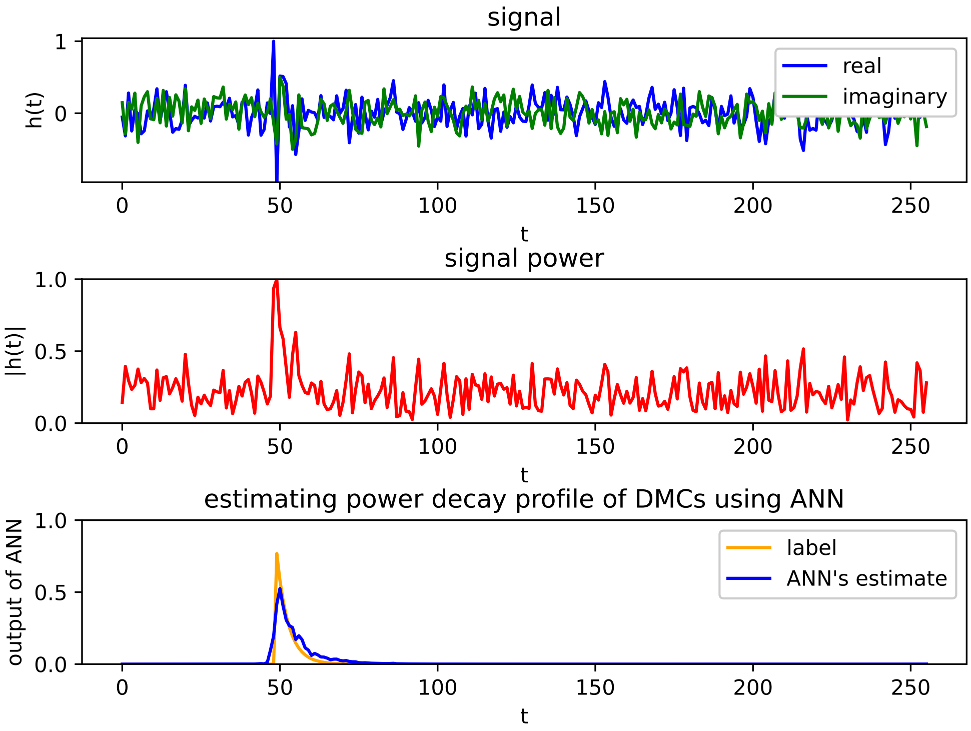

Figure 7.9: Input-output example of ANN estimating PDP of DMCs. The t is a number of ANN's input-output which corresponds to time delay normalized to sampling period.

7.4.1 Architecture

The very same architecture as the one which we have introduced when roughly estimating ToAs of MPCs (previous section) is used to estimate power delay profile of DMCs. Architecture diagram in fig. 7.13.

Inspired by pattern recognition oriented ANNs, we constructed our ANN as a sequential model containing several layers of 1-D convolutions together with Leaky ReLU activation functions followed by one dense layer with Sigmoid activation function at its output.

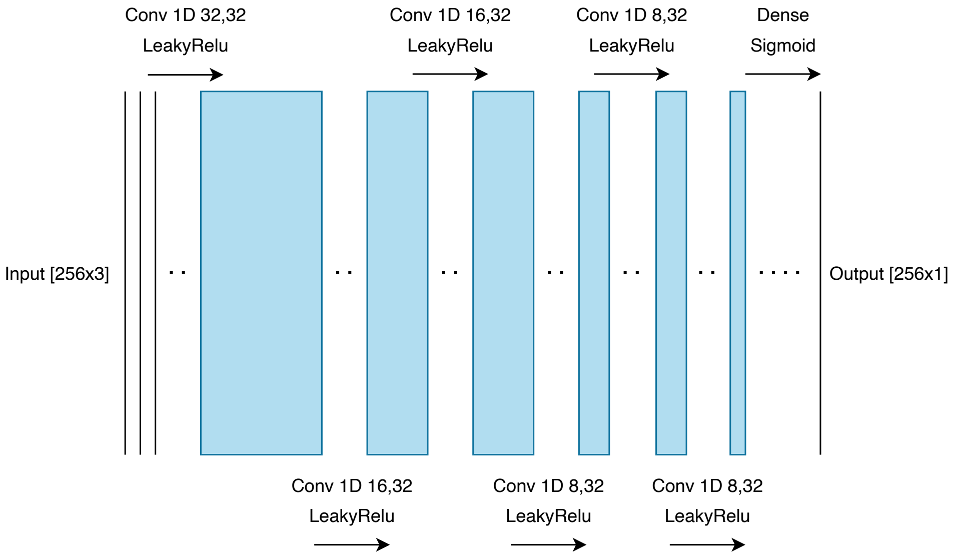

Figure 7.10: ANN estimating DMCs.

7.4.2 Train and test dataset

We have generated train and test datasets specified in tab. 7.1 and trained the network using Adam optimizer [15] which has been proven as a well working complex optimizing method for training of broad field of ANNs and binary cross-entropy as loss function.

parameter	value
`dataset_n`	100000 for train and 10000 for test
`sample_l`	256
`mpc_n`	$\{1 \ldots 3\}$
`mpc_snr_base`	0
`mpc_snr_var`	20
`dmc_trigger`	True
`dmc_power_base`	0.25
`dmc_power_var`	0.5
`dmc_len_base`	16
`dmc_len_var`	32
`awgn_trigger`	True

Table 7.2: List of channel impulse response simulator's input parameters used to train ANN estimating power decay profile of DMCs.

7.4.3 Performance

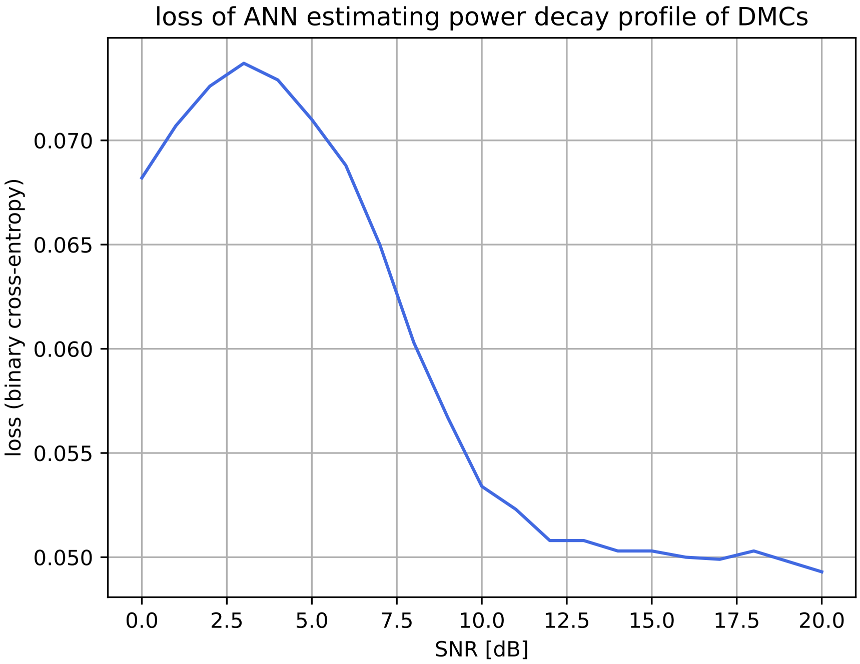

Figure 7.11: Loss of ANN estimating power decay profile of DMCs.

Loss showed in fig 7.11 does not say much information. Output of this ANN is later fed in another ANN where the performance is measured. There is no intuitive way of measure the performance of this ANN apart of its loss. You can see that loss naturally decreases while SNR increases since the value SNR[dB] ≈ 3. The weird part is that the worst loss in not in location of the lowest SNRs. Our interpretation of this local maximum of loss when $0 \leq$ SNR[dB] ≤ 3 is that it is so hard for the ANN to estimate PDPs of DMCs since the noise is likely of higer power than DMCs that the output converges to universal output values no matter the given input.

Because of the very same architecture as the one used while roughly estimating ToAs of MPCs we assume the very same principle of performance for various number of MPCs and DMCs hold true and is following: to achieve the best performance on unknown number of MPCs and DMCs one should train the ne

Because output of this ANN is mainly used as an input to ANN fine estimating ToAs of MPCs introduced in later section.

7.5 ANN fine estimating ToAs

In this section we introduce ANN which fine estimates ToAs ($\boldsymbol{\tau}$) of MPCs. The ANN takes real and imaginary components as inputs together with computed signal power (2.20) and outputs of ANNs used to roughly estimate ToAs of MPCs and PDPs of DMCs. The output is a scalar number τ^{ann} normalized to interval from 0 to 1 which is estimate of ToA of MPC. Input-output example of the ANN in fig. 7.12.

We do not feed the whole CIR sample as an input. Using rough estimation of peak position (ToA of MPC) by previously introduced ANN, we crop a window of length 64 from the original CIR and two outputs of ANNs estimating rough peak location and PDP of DMCs. We select the window in such a way that the expected peak τ^{rough} is located at random position uniformly and fulfills $14 \leq \tau^{\mathrm{rough}} \leq 18$ - that means uniformly within two samples of the one quarter of the selected window. One quarter because the DMCs carry the valuable information of ToAs of MPCs and they are located after the main peak. Samples before the main peak only carry information about power of AWGN.

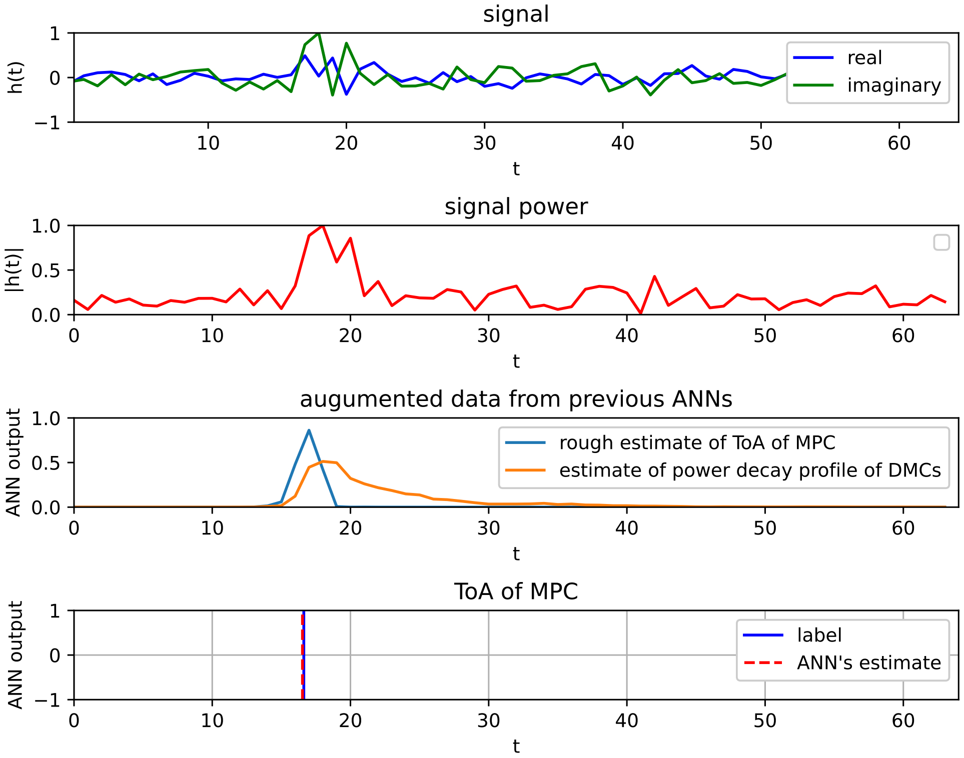

Figure 7.12: Input-output example of ANN fine estimating ToA of MPC. The t is a number of ANN's input-output which corresponds to time delay normalized to sampling period.

7.5.1 Architecture

Inspired by [12] we constructed our ANN as a sequential model containing several layers of 1-D convolutions together with Leaky ReLU activation functions followed by several dense layers with Sigmoid activation function at their outputs.

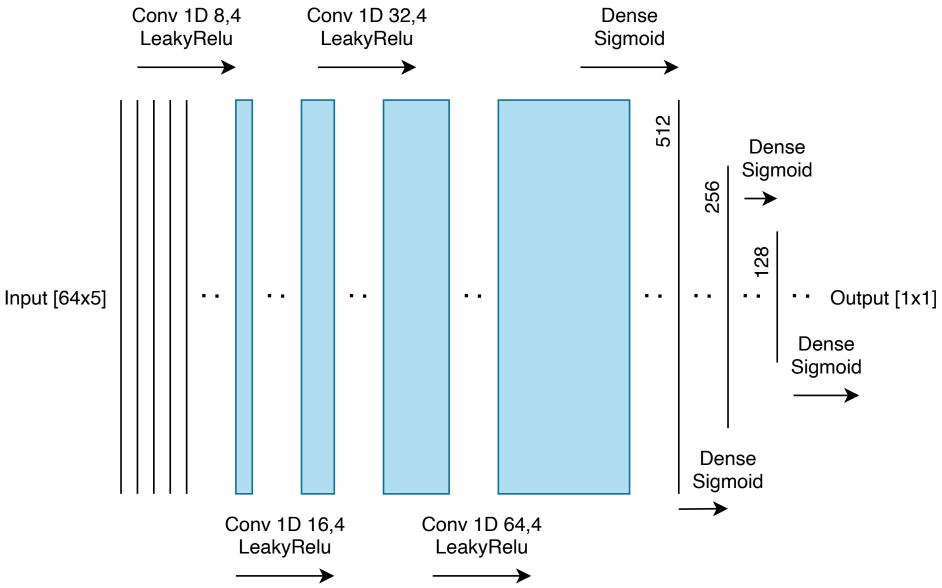

Figure 7.13: ANN estimating fine ToA of MPCs.

7.5.2 Train and test dataset

We have generated train and test datasets specified in tab. 7.1 and trained the network using Adam optimizer [15] which has been proven as a well working complex optimizing method for training of broad field of ANNs and binary cross-entropy as loss function.

parameter	value
`dataset_n`	100000 for train and 10000 for test
`sample_l`	256
`mpc_n`	1 - ANN focuses on one particular MPC
`mpc_snr_base`	0
`mpc_snr_var`	20
`dmc_trigger`	True
`dmc_power_base`	0.25
`dmc_power_var`	0.5
`dmc_len_base`	16
`dmc_len_var`	32
`awgn_trigger`	True

Table 7.3: List of channel impulse response simulator's input parameters used to train ANN fine estimating ToAs of DMCs.

7.5.3 Performance

Since the output of the ANN is an estimate of ToA of a particular MPC as a scalar number, we can compare ANN's performance in terms of RMSE to the theoretical lower bound - CRLB and reference SAGE algorithm.

We present results of series of evaluation on datasets that posses various SNRs in fig. 7.14.

It is clear that RMSE of both: our approach using ANN and reference SAGE algorithm fall with rising SNR. Significant success of our approach is that around the value SNR [db] ≈ 7.5 performance of both are even and since this point our proposed method perform better.

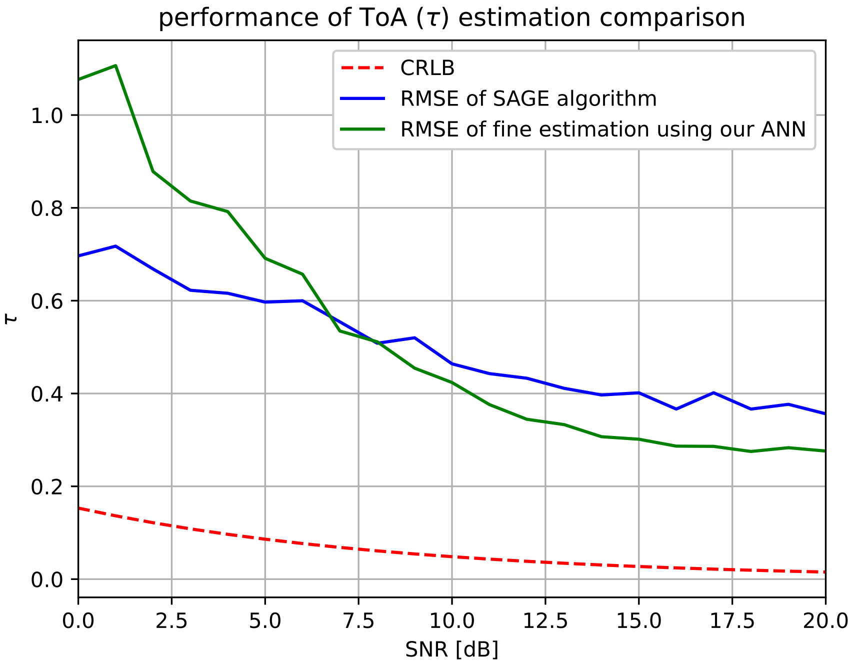

Figure 7.14: Performance of ToA (τ) estimation comparison. τ is normalized to sampling period.

This ANN strongly relies on quality of outputs of previously introduced ANNs that serve us rough estimator of MPCs and PDP of DMCs. It is not possible to provide good estimate of PDP of DMCs when the SNR is getting close to 0 because power of AWGN tops the power of DMCs in majority of samples.

Chapter 8

Conclusion

The goal of our work was to explore opportunities the ANNs could offer to the field of wireless parameter estimation. There is a need for a method capable of real-time wireless channel identification in order to improve performance of positioning algorithms that are restricted by the precision of estimated wireless channel parameters. Computational complexity of state-of-the-art methods are not allowing real-time processing required for positioning.

For purposes of our theoretical research it is necessary to have datasets containing CIRs of desired parameters and their precise ground truth labels. It is not possible to obtain such samples by real measurement. This implies need of development of a CIR capable of both correctly providing synthetic artificial channel measurements and respecting limits of measurement equipment in order to obtain realistic signals at the same time. Testing on real data is subject of future work. We first theoretically and algebraically described channel model that consists of MPCs, DMCs and AWGN with all its limitations and later developed the channel impulse response simulator based on it.

CRLB was introduced as a theoretical lower bound on ToA estimation together with SAGE algorithm as a ToA estimation performance reference. A few drawbacks come with SAGE algorithm. Starting with its high computational complexity, need of apriory knowledge of number MPCs and providing initial estimates. We introduce method to estimate DMC parameters using ANN, utilizing these estimations to improve performance of reference SAGE algorithm is subject of future work. Field of our research does not reach beyond the original SAGE algorithm which we implemented, tested and used as reference algorithm. It is possible to recover peak's ToA further than 3x beyond the CIR's resolution for SNR $\geq 14\,\mathrm{dB}$.

Our proposed method is based on two-level processing using ANNs when we first roughly estimate ToA of MPCs and PDP of DMCs using two ANNs and feed the cropped windows of these estimates as inputs into the second-level ANN which fine estimates ToAs of MPCs. Our approach decomposes estimation of several MPCs into a separate problems which are all done in constant time (ANN evaluation) - suitable for real-time applications. It takes up to 10 ms to fine esimate ToAs of MPCs using proposed ANN in comparison to more than 250 ms using SAGE on mainstream laptop. We state that for SNR $\geq$ 7.5 dB our proposed method perform~~s~~ better estimation~~s~~ of ToAs of MPCs (in terms of RMSE) that the reference SAGE algorithm.

To summarize our results: we have proposed ANN approach to fine estimate ToAs of MPCs in discrete-sampled noised CIR. Our method not only outperforms reference SAGE algorithm for SNR $\geq$ 7.5 dB in terms of estimation's RMSE but also addresses limitations and problems that come along with the SAGE that are: initial guessing of number of MPCs and necessary initial estimates of ToAs of MPCs. Additional application of estimating PDP of DMCs is to feed the information to state-of-the-art methods to improve their performance.

Bibliography

[1] P. Bello. Time-frequency duality. *IEEE Transactions on Information Theory*, 10(1):18–33, 1964.

[2] E. J. Candès and C. Fernandez-Granda. Towards a mathematical theory of super-resolution. *CoRR*, abs/1203.5871, 2012.

[3] H. Cramer. *Mathematical methods of statistics / by Harald Cramer*. Princeton University Press Princeton, 1946.

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[5] V. Erceg, D. G. Michelson, S. S. Ghassemzadeh, L. J. Greenstein, A. J. Rustako, P. B. Guerlain, M. K. Dennison, R. S. Roman, D. J. Barnickel, S. C. Wang, and R. R. Miller. A model for the multipath delay profile of fixed wireless channels. *IEEE Journal on Selected Areas in Communications*, 17(3):399–410, 1999.

[6] J.A. Fessler and A.O. Hero. Space-alternating generalized expectation-maximization algorithm. *IEEE Transactions on Signal Processing*, 42(10):2664–2677, 1994.

[7] B. H. Fleury, M. Tschudin, R. Heddergott, D. Dahlhaus, and K. Ingeman Pedersen. Channel parameter estimation in mobile radio environments using the sage algorithm. *IEEE Journal on Selected Areas in Communications*, 17(3):434–450, 1999.

[8] J.M. Francos and B. Friedlander. Bounds for estimation of complex exponentials in unknown colored noise. *IEEE Transactions on Signal Processing*, 43(9):2176–2185, 1995.

[9] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

[10] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[11] J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79:2554–8, 05 1982.

[12] Y. S. Hsiao, M. Yang, and H. S. Kim. Super-resolution time-of-arrival estimation using neural networks. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 1692–1696, 2021.

[13] Thomas Jost, Wei Wang, Uwe-Carsten Fiebig, and Fernando Perez Fontan. Detection and tracking of mobile propagation channel paths. *IEEE Transactions on Antennas and Propagation*, 60:4875–4883, 10 2012.

[14] S. M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice Hall, 1997.

[15] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[16] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[17] W. Mcculloch and W. Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:127–147, 1943.

[18] A. V. Oppenheim, A. S. Willsky, and S. H. Nawab. *Signals amp; Systems (2nd Ed.)*. Prentice-Hall, Inc., USA, 1996.

[19] K. Pahlavan and A. H. Levesque. *Wireless Information Networks (Wiley Series in Telecommunications and Signal Processing)*. John Wiley Sons, Inc., Hoboken, NJ, USA, 2005.

[20] K. I. Pedersen, P. E. Mogensen, and B. H. Fleury. A stochastic model of the temporal and azimuthal dispersion seen at the base station in outdoor propagation environments. *IEEE Transactions on Vehicular Technology*, 49(2):437–447, 2000.

[21] A. Richter. Parametric modelling and estimation of distributed diffuse scattering components of radio channels. 2003.

[22] A. Richter. *Estimation of radio channel parameters*. ISLE, 2005.

[23] E. Saff and A. Snider. *Fundamentals of Complex Analysis with Applications to Engineering, Science, and Mathematics*, pages –563. 01 2003.

[24] S. Salous and R. Thoma. State of the art mobile radio channel sounding and data analysis. In *2014 XXXIth URSI General Assembly and Scientific Symposium (URSI GASS)*, pages 1–3, 2014.

[25] P. Stoica, A. Jakobsson, and Jian Li. Cisoid parameter estimation in the colored noise case: asymptotic cramer-rao bound, maximum likelihood, and nonlinear least-squares. *IEEE Transactions on Signal Processing*, 45(8):2048–2059, 1997.

[26] K. Strachnyi. Brief history of neural networks, Jan 2019.

[27] A. Zangwill. *Modern electrodynamics.* Cambridge Univ. Press, Cambridge, 2013.

[28] D. Čoja, N. Nešković, and A. Nešković. Channel impulse response estimation using vector network analyzer. In *2017 25th Telecommunication Forum (TELFOR)*, pages 1–4, 2017.

Appendices

Appendix A

Acronyms

ANN artificial neural network

AWGN additive white Gaussian noise

BLUE best linear unbiased estimator

CIR channel impulse response

CNN convolution neural network

CRLB Cramér-Rao Lower Bound

DMC dense multi-path component

EM expectation–maximization

FT Fourier transformation

IFT inverse Fourier transformation

Leaky ReLU leaky rectified linear unit

LSTM Long Short-Term Memory

ML maximum likelihood

MLE maximum likelihood estimate

MPC multi-path component

PDP power delay profile

ReLU rectified linear unit

RMSE root mean square error

RNN recurrent neural network

SAGE space-alternating generalized expectation-maximization

SNR signal to noise ratio

SNR signal to noise ratio

ToA time of arrival

Appendix B

Contents of the attachment

- `thesis/` - thesis in PDF format
- `code/` - Python code needed to replicate our work