Applications of Deep Learning to the Design of Enhanced Wireless Communication Systems

Mathieu Goutay

▶ **To cite this version:**

Mathieu Goutay. Applications of Deep Learning to the Design of Enhanced Wireless Communication Systems. Information Theory [cs.IT]. Université de Lyon, 2022. English. NNT : . tel-03656696

HAL Id: tel-03656696
https://hal.science/tel-03656696

Submitted on 2 May 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

NOKIA BELL LABS – UNIVERSITY OF LYON
Doctoral School of Electronics, Electrotechnics and Automation
(ED EEA)

THESIS

presented on January 28, 2022 by

Mathieu Goutay

for the degree of

Doctor of Philosophy

Specialization: Signal and Image Processing

Applications of Deep Learning to the Design of Enhanced Wireless Communication Systems

---

Members of the Jury:

Thesis supervisor:		
Gorce, Jean-Marie	Professor	University of Lyon, France
Thesis co-supervisors:		
Hoydis, Jakob	Principal Research Scientist	Nvidia, France*
Ait Aoudia, Fayçal	Senior Research Scientist	Nvidia, France*
		*Previously at Nokia Bell Labs, France
Reviewers:		
Le Ruyet, Didier	Professor	CNAM, France
Langlais, Charlotte	Permanent Research Staff, HDR	IMT Atlantique, France
Examiners:		
Fijalkow, Inbar	Professor	ENSEA, France
ten Brink, Stephan	Professor	University of Stuttgart, Germany

NOKIA BELL LABS – UNIVERSITÉ DE LYON
École Doctorale d'Électronique, Électrotechnique et Automatique
(ED EEA)

THÈSE

présentée le 28 janvier 2022 par

Mathieu Goutay

en vue de l'obtention du

Doctorat de l'Université de Lyon

Spécialité : Traitement du Signal et de l'Image

Applications de l'Apprentissage Profond à la Conception de Systèmes de Communication Sans Fil Améliorés

---

Devant le jury composé de :

Directeur de thèse :		
Gorce, Jean-Marie	Professeur	Université de Lyon, France
Co-encadrants de thèse :		
Hoydis, Jakob	Chargé de Recherche	Nvidia, France*
Ait Aoudia, Fayçal	Chargé de Recherche	Nvidia, France*
		*Précéd. à Nokia Bell Labs, France
Rapporteurs :		
Le Ruyet, Didier	Professeur	CNAM, France
Langlais, Charlotte	Chargée de Recherche, HDR	IMT Atlantique, France
Examinateurs :		
Fijalkow, Inbar	Professeure	ENSEA, France
ten Brink, Stephan	Professeur	Université de Stuttgart, Allemagne

Acknowledgements

First, I would like to express my deepest appreciation to Dr. Jean-Marie Gorce, who has guided me through the last six years of my studies, to Dr. Jakob Hoydis, who gave my this incredible Ph.D. opportunity three years ago and has always inspired me since, and to Fayçal Ait Aoudia, for his invaluable help, motivation, and support. It has been a great privilege and honor to work under your shared supervision. In addition, I would like to thank all my colleagues who have contributed to creating a fantastic work environment.

Secondly, I am extremely grateful to my parents Bernard and Marie-Hélène, and to my stepfather Fabrice, for their constant love and affection. The education I received is one of my greatest assets to navigate this world, and so I would like to dedicate this manuscript to you. Special thanks to my brother Nicolas, for showing me the way and proving me that everything was possible after high school. Thanks also to my grandparents, Jeanne-Marie, Joseph, Monique, and Robert, for always being there for me.

Finally, I would like to express my gratitude to my girlfriend, Camille, who has always believed in me. Your love, humor and support have been a steady light in the lonely moments of the pandemic. I would also like to give warm thanks all my friends, starting with *les anciens*, who have been with me for so many years, and all the other friends from INSA and elsewhere who helped me grow and become who I am.

Abstract

Innovation in the physical layer of communication systems has traditionally been achieved by breaking down the transceivers into sets of processing blocks, each optimized independently based on mathematical models. This approach is now challenged by the ever-growing demand for wireless connectivity and the increasingly diverse set of devices and use-cases. Conversely, deep learning (DL)-based systems are able to handle increasingly complex tasks for which no tractable models are available. By learning from the data, these systems could be trained to embrace the undesired effects of practical hardware and channels instead of trying to cancel them. This thesis aims at comparing different approaches to unlock the full potential of DL in the physical layer.

First, we describe a neural network (NN)-based block strategy, where an NN is optimized to replace one or multiple block(s) in a communication system. We apply this strategy to introduce a multi-user multiple-input multiple-output (MU-MIMO) detector that builds on top of an existing DL-based architecture. The key motivation is to replace the need for retraining on each new channel realization by a hypernetwork that generates optimized sets of parameters for the underlying DL detector. Second, we detail an end-to-end strategy, in which the transmitter and receiver are modeled as NNs that are jointly trained to maximize an achievable information rate. This approach allows for deeper optimizations, as illustrated with the design of waveforms that achieve high throughputs while satisfying peak-to-average power ratio (PAPR) and adjacent channel leakage ratio (ACLR) constraints. Lastly, we propose a hybrid strategy, where multiple DL components are inserted into a traditional architecture but trained to optimize the end-to-end performance. To demonstrate its benefits, we propose a DL-enhanced MU-MIMO receiver that both enable lower bit error rates (BERs) compared to a conventional receiver and remains scalable to any number of users.

Each approach has its own strengths and shortcomings. While the first one is the easiest to implement, its individual block optimization does not ensure the overall system optimality. On the other hand, systems designed with the second approach are computationally complex and do not comply with current standards, but allow the emergence of new opportunities such as high-dimensional constellations and pilotless transmissions. Finally, even if the block-based architecture of the third approach prevents deeper optimizations, the combined flexibility and end-to-end performance gains motivate its use for short-term practical implementations.

Résumé

L'innovation dans la couche physique des systèmes de communications a traditionnellement été réalisée en modélisant les émetteurs-récepteurs comme une suite de blocs, chacun étant optimisé indépendamment sur la base de modèles mathématiques. Cette approche est aujourd'hui remise en question par la demande croissante de connectivité et la diversité des cas d'utilisation. À l'inverse, les systèmes basés sur l'apprentissage profond (deep learning, DL) sont capables de traiter des tâches de plus en plus complexes en apprenant à partir de données. Cette thèse vise donc à comparer différentes approches pour exploiter le plein potentiel du DL dans la couche physique.

Tout d'abord, nous décrivons une stratégie basée sur un réseau neuronal (neural network, NN) qui est optimisé pour remplacer un ou plusieurs blocs consécutifs dans un système de communication. Nous appliquons cette stratégie pour introduire un détecteur multi-utilisateurs à entrées et sorties multiples (multi-user multiple-intput multiple-output, MU-MIMO) qui s'appuie sur un détecteur existant basé sur du DL. L'idée est d'utiliser un hyper-réseau de neurones pour générer des parèmetres optimisés pour le détecteur DL sous-jacent. Deuxièmement, nous détaillons la stratégie de bout en bout, dans laquelle les émetteurs-récepteurs sont modélisés comme des NNs qui sont entraînés conjointement pour maximiser un taux d'information réalisable. Cette approche permet des optimisations plus profondes, comme l'illustre la conception de formes d'onde qui atteignent des débits élevés tout en satisfaisant des contraintes sur le signal et son spectre. Enfin, nous proposons une stratégie hybride, où plusieurs composants DL sont insérés dans une architecture traditionnelle mais entraînés pour optimiser les performances de bout en bout. Pour démontrer ses avantages, nous proposons un récepteur MU-MIMO amélioré par DL qui permet à la fois de réduire les taux d'erreur binaire (bit error rate, BER) par rapport à un récepteur classique et de rester adaptable à un nombre variable d'utilisateurs.

Chaque approche a ses propres forces et faiblesses. Si la première est la plus facile à implémenter, l'optimisation individuelle de chaque bloc ne garantit pas l'optimalité du système entier. En revanche, les systèmes conçus selon la seconde approche sont souvent trop complexes et ne sont pas conformes aux standards actuels, mais ils permettent l'émergence de nouvelles possibilités telles que des constellations de grande dimension et des transmissions sans pilote. Enfin, même si l'architecture par blocs de la troisième approche empêche des optimisations plus poussées, la combinaison de sa flexibilité et de son optimisation de bout en bout motive son utilisation pour des implémentations à court terme.

Un résumé de la thèse en français est disponible dans l'Appendix B de ce manuscript.

Publications

Journal papers published during my Ph.D. studies

[1] M. Goutay, F. Ait Aoudia, J. Hoydis, and J.-M. Gorce, "Machine Learning for MU-MIMO Receive Processing in OFDM Systems", *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2318–2332, 2021. DOI: 10.1109/JSAC.2021.3087224.

[2] M. Goutay, F. Ait Aoudia, J. Hoydis, and J.-M. Gorce, "Learning OFDM Waveforms with PAPR and ACLR Constraints", 2021. arXiv: 2110.10987 [cs.IT].

Conference papers published during my Ph.D. studies

[1] M. Goutay, F. Ait Aoudia, J. Hoydis, and J.-M. Gorce, "End-to-End Learning of OFDM Waveforms with PAPR and ACLR Constraints", *Accepted at 2021 IEEE Global Communication Conference (GLOBECOM)*, 2021. arXiv: 2106.16039 [cs.IT].

[2] M. Goutay, F. Ait Aoudia, J. Hoydis, and J.-M. Gorce, "Machine Learning-enhanced Receive Processing for MU-MIMO OFDM Systems", *Accepted at 2021 IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2021. arXiv: 2106.16074 [cs.IT].

[3] M. Goutay, F. Ait Aoudia, and J. Hoydis, "Deep HyperNetwork-Based MIMO Detection", in *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2020, pp. 1–5. DOI: 10.1109/SPAWC48557.2020.9154283.

Conference papers published during my master studies

[1] M. Goutay, F. Ait Aoudia, and J. Hoydis, "Deep Reinforcement Learning Autoencoder with Noisy Feedback", in *2019 International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)*, 2019, pp. 1–6. DOI: 10.23919/WiOPT47501.2019.9144089.

[2] M. Goutay, L. S. Cardoso, and C. Goursaud, "Massive Machine Type Communications Uplink Traffic: Impact of Beamforming at the Base Station", in *2018 25th International Conference on Telecommunications (ICT)*, 2018, pp. 493–497. DOI: 10.1109/ICT.2018.8464894.

Patents and Tutorials

Patents granted at the time of publication

[1] M. Goutay, F. Ait Aoudia, and J. Hoydis, *A receiver for a communication system*, 2021. Patent number: `WO2021151477A1`.

[2] M. Goutay, F. Ait Aoudia, and J. Hoydis, *Apparatuses and methods for providing feedback*, 2021. Patent number: `WO2021083521A1`.

[3] M. Goutay, F. Ait Aoudia, and J. Hoydis, *Communication system having a configurable modulation order and an associated method and apparatus*, 2020. Patent number: `US2020403723A1`.

Tutorials

[1] M. Goutay. "Machine learning for the physical layer (lecture materials, project and exercises)". (2021), [Online]. Available: `https://mgoutay.github.io/tutorials/2020/02/12/ml_courses.html` (visited on 11/08/2021).

[2] M. Goutay. "Tensorflow 2.0 tutorial". (2019), [Online]. Available: `https://mgoutay.github.io/tutorials/2019/11/14/tf2-tutorial.html` (visited on 11/08/2021).

Contents

— 1 —

Introduction

1.1 When Machine Learning Meets Signal Processing

The first neural network (NN) model was introduced in 1943 [1], but sixty years of research and of processing power increase were required to enable a major adoption of machine learning (ML) by the industry [2]. In particular, the 2010s have seen significant improvements in parallel computing, leading to the advent of deep learning (DL) and to breakthroughs in computer vision [3], speech recognition [4], and many other domains [5]–[7]. DL is especially useful when the task at hand is difficult to formalize mathematically or when the mathematical models are untractable. By shifting from model-driven to data-driven algorithms, DL techniques are able to circumvent this problem as long as a sufficient dataset is available. Typically, massive progresses have been possible in the field of image recognition thanks to the publication of the ImageNet dataset in 2009 [8], containing more than 3 millions labelled images.

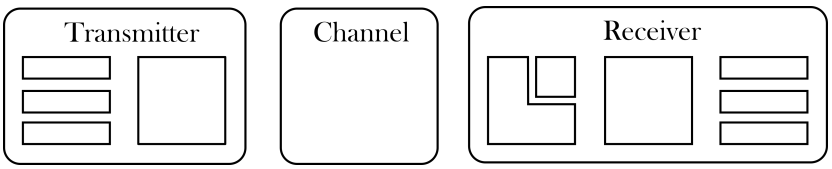
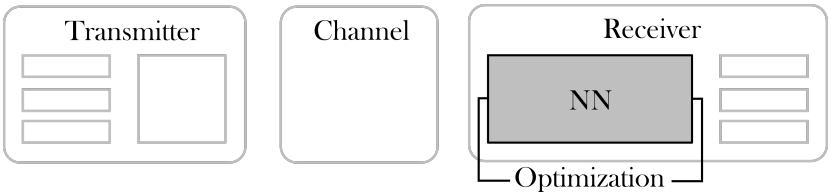


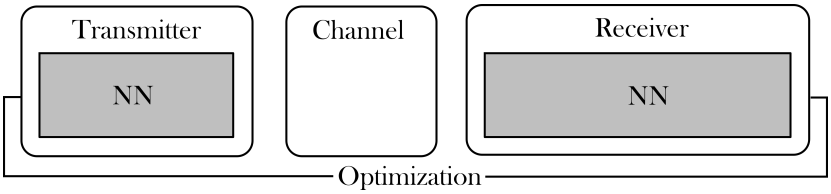
Figure 1.1: A traditional block-based communication system.

In the meantime, new generations of cellular communication systems have emerged every ten years, starting from 1979 [9]. Each generation brings multiple connectivity improvements, such as faster and more reliable communications, in part thanks to a better modeling of the wireless channel. These mathematical models allowed the design of algorithms that can take advantage of the available knowledge in information theory and signal processing. As the transmitters and receivers became more and more complex, tractability was achieved by splitting the transmit and receive processing chains into small components, usually referred to

as *processing blocks* and illustrated in Fig. 1.1. Such bloc-based communication systems suffer from multiple drawbacks. On the one hand, simplistic channel models fails at capturing all the specificities of the underlying hardware and propagation phenomenons. On the other hand, the joint optimization of the transmitter and receiver quickly becomes intractable when more realistic channels models are derived, and therefore the optimization of each block is typically performed independently. This does not ensure the optimality of the resulting system, as it can be shown for the channel coding and modulation blocks [10]. Finally, signalling is often required between the transmitter and the receiver, which introduces an overhead that reduces the system throughput.



(a) NN-based block optimization: an NN is optimized to replace one or multiple block(s) in a communication system.



(b) End-to-end optimization: NN-based transceivers are optimized to maximize the end-to-end performance of a system.

Figure 1.2: Different level of NN integrations into communication systems.

DL for the physical layer was already studied in the nineties [11], but a renewed interest started in 2016-1017 thanks to the publication of multiple promising papers. One of them was published in 2016 by Be'ery et al., who represented the channel decoding algorithm as an NN to improve the bit error rates (BERs) of systems using various codes [12]. This approach corresponds to an *NN-based block optimization strategy* as shown in Fig. 1.2a, in which one or multiple consecutive processing blocks are replaced by an NN. To handle such NN-based blocks, the next generation of communication system needs to be *designed for DL*, in a way that allows for practical training and testing of these components. Although this idea is interesting, the true DL revolution started when O'Shea and Hoydis introduced end-to-end learning for communication systems in their seminal paper from 2017 [13]. Such approach is often referred to the *end-to-end optimization strategy*, as depicted in Fig. 1.2b, and experimental gains were quickly shown by Hoydis, Dörner, Cammerer et al. with over-the-air transmissions [14]. This strategy allows the systems to be entirely optimized from real-world data, therefore enabling efficient handling of hardware impairments and other channel distortions without requiring any mathematical model [15]. Moreover, they can contribute to reducing the signaling overhead,

either by removing the pilots required for channel estimation at the receiver [16] or by learning optimal medium access control (MAC) protocols [17]. For these reasons, end-to-end systems are often seen as the next big step in the evolution of the physical layer, as the transmit and receive processing would be *designed by DL* [18].

The study of the two strategies presented in Fig. 1.2 have lead to the discovery of deep connections between the fields of DL, information theory, and of signal processing in communication systems [19]. For example, the transmitter-receiver pair can be modeled as an autoencoder, in which the estimation of the transmitted bits becomes a binary classification problem [20]. Moreover, an achievable rate of a communication system can be expressed in terms of cross entropy [21], which is a well-known metric in information theory and DL. These connections, along with performance improvements demonstrated on multiple systems and environments, indicate that DL will play an important role in future generations of communication systems [22]–[24]. However, each strategy has its shortcomings: the block-based NNs (Fig. 1.2a) are not trained to maximize the overall performance of the system, and the fully learned transceivers (Fig. 1.2b) lack interpretability and scalability. This thesis therefore aims at providing some answers to the question of the optimal integration of DL components into wireless communication systems.

1.2 Current Challenges and Contributions of this Work

The next generation of cellular networks will need to support a growing number of different services and devices [25]. To that aim, the available resources need to be more efficiently shared among users. One key technique is the use of multi-user multiple-input multiple-output (MU-MIMO) systems, where spatial multiplexing is exploited to increase both the channel capacity and the number of users that can be served simultaneously [26]. One of the main challenges related to the deployment of such systems is the complexity of the symbol detection algorithm, which grows with the number of antennas and users. For example, maximum a posteriori (MAP) detection is optimal but known to be NP-hard, and sphere decoders have exponential worst-case complexity [27]. The conventional solution to tackle this problem is to use linear detectors that are computationally tractable, but suffer from performance degradation on ill-conditioned channels. In the past years, several approaches tried to address those challenges by implementing the detector as an NN, which corresponds to the NN-based block optimization strategy. However, they either still achieve unsatisfying performance on spatially correlated channels, or are computationally demanding since they require retraining for each channel realization. In this work, we address both issues by training an additional NN, referred to as the hypernetwork, which takes as input the channel matrix and generates the weights of the NN-based detector. Results show that the proposed approach achieves near state-of-the-art performance without the need for re-training.

Another key research direction is the improvement of the orthogonal frequency-division multiplexing (OFDM) waveform, used in most modern communication systems such as 4G,

5G, and Wi-Fi. Indeed, conventional OFDM suffers from multiple drawbacks, such as a high peak-to-average power ratio (PAPR) and adjacent channel leakage ratio (ACLR). To tackle these problems, we leverage the end-to-end optimization strategy and model the transmitter and receiver as NNs that respectively implement a high-dimensional modulation scheme and estimate the transmitted bits. We then propose a learning-based approach to design OFDM waveforms that satisfy selected constraints while maximizing an achievable information rate, with the additional advantage that no pilots are needed during transmissions. Evaluated with ACLR and PAPR targets, the trainable system is able to satisfy the constraints while enabling significant throughput gains compared to a tone reservation (TR) baseline.

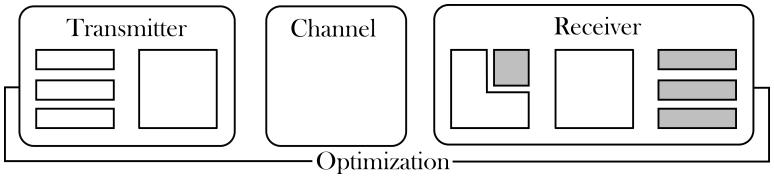


Figure 1.3: A hybrid training strategy.

As mentioned in Section 1.1, such end-to-end systems lack interpretability, as the black-box design prevents the capture of intermediate data such as channel estimates. They also lack scalability, which is especially important in MU-MIMO transmissions since the receive algorithm must allow for easy adaptation to a varying number of users. It is therefore still unclear if this strategy is competitive with respect to conventional MU-MIMO receivers in realistic scenarios and under practical constraints. For this reason, we propose a DL-enhanced MU-MIMO receiver that builds on top of a conventional architecture to preserve its interpretability and scalability, but is trained to maximize an achievable transmission rate. This approach can be seen as a *hybrid strategy*, in which multiple DL-based components are inserted in a traditional block-based architecture but are optimized to maximize the end-to-end system performance (Fig. 1.3). The resulting system can be used in the up- and downlink and does not require hard-to-get perfect channel state information (CSI) during training, which contrasts with existing works. Simulation results demonstrate consistent performance improvements over a linear minimum mean squared error (LMMSE) baseline which are especially pronounced in high mobility scenarios.

1.3 Thesis Outline

The remainder of this thesis is organized as follows. Chapter 2 provides a background on DL, on the physical layer, and on the interconnection between the two fields. OFDM is presented first, with a derivation of the channel models corresponding to both single-input single-output (SISO) and MU-MIMO transmissions. We then detail the concept of backpropagation, NNs, and of stochastic gradient descent (SGD), and describe the optimization of DL-enhanced systems. Chapter 3 introduces the hypernetwork-based MIMO detector, which is an example of the block-based optimization strategy depicted in Fig. 1.2a. The traditional iterative

detection framework and the concept of hypernetworks are presented, followed by a description of the HyperMIMO system and of evaluations on spatially correlated channels. The fully NN-based transceiver strategy of Fig. 1.2b is discussed in Chapter 4, where we design OFDM waveforms that both maximize an achievable rate and satisfy PAPR and ACLR constraints. The system model and the baseline are described, and both the NN architectures and the learning-based approach used for waveform design are detailed. Finally, simulation results and insights are provided. Chapter 5 is dedicated to the presentation of the hybrid strategy (Fig. 1.3) for DL-enhanced MU-MIMO receivers. First, we develop the traditional architectures corresponding to uplink and downlink transmissions. Second, we highlight two limitations of these architectures and detail how we address them using convolutional neural networks (CNNs). Third, we provide simulation results to compare the DL-enhanced receiver to the baseline. Finally, Chapter 6 concludes this manuscript.

1.4 Notations

$\mathbb{R}$ ($\mathbb{C}$) denotes the set of real (complex) numbers. Tensors and matrices are denoted by bold upper-case letters and vectors are denoted by bold lower-case letters. We respectively denote by $\mathbf{m}_a$ and $m_{a,b}$ the vector and scalar formed by slicing the matrix $\mathbf{M}$ along its first and second dimensions. Note that the notation $[\mathbf{M}]_a$ and $[\mathbf{M}]_{a,b}$ is also used in Section 2.2.2 for clarity. Similarly, we denote by $\mathbf{T}_{a,b} \in \mathbb{C}^{N_c \times N_d}$ ($\mathbf{t}_{a,b,c} \in \mathbb{C}^{N_d}$, $t_{a,b,c,d} \in \mathbb{C}$) the matrix (vector, scalar) formed by slicing the tensor $\mathbf{T} \in \mathbb{C}^{N_a \times N_b \times N_c \times N_d}$ along the first two (three, four) dimensions. The notation $\mathbf{T}^{(k)}$ indicates that the quantity at hand is only considered for the k^{th} user, and $\mathbf{v}_{-a}$ corresponds to the vector $\mathbf{v}$ from which the a^{th} element was removed. $||\mathbf{M}||_{\text{F}}$ denotes the Frobenius norm of $\mathbf{M}$. Card($\mathcal{S}$) denotes the number of elements in a set $\mathcal{S}$, vec $(\cdot)$ the vectorization operator, and $\odot$ and $\oslash$ the element-wise product and division, respectively. $(\cdot)^{\mathsf{T}}$, $(\cdot)^{\mathsf{H}}$, and $(\cdot)^{\star}$ respectively denote the transpose, conjugate transpose, and element-wise conjugate operator. $I(\mathbf{x};\mathbf{y})$ and $P(\mathbf{x},\mathbf{y})$ represent the mutual information and joint conditional probability of $\mathbf{x}$ and $\mathbf{y}$, respectively. $\mathbf{I}_N$ is the $N \times N$ identity matrix and $\mathbb{1}_{N \times M}$ is the $N \times M$ matrix where all elements are set to 1. Finally, the imaginary unit is j, such that $j^2 = -1$.

— 2 —

Background on the Physical Layer and Deep Learning

2.1 OFDM Systems

A digital communication system aims at transmitting bits from a transmitter to a receiver by modulating an electromagnetic wave that is transmitted through a channel (Fig. 2.1). Multiple waveforms can be used to carry the information, and the waveform choice is usually dictated by the channel distortions that need to be dealt with. In the following, we present orthogonal frequency-division multiplexing (OFDM), a transmission technique used in most modern communication systems thanks to its ability to handle difficult channel conditions such as selective fading.

Figure 2.1: A digital communication system.

2.1.1 Transmit Processing

The first operation that is performed by the transmitter is the *bit mapping*, in which vectors of Q bits $\mathbf{b} \in \{0,1\}^Q$ are mapped to 2^Q different *symbols* $x \in \mathcal{C}$, where $\mathcal{C}$ is referred to as the constellation. Quadrature amplitude modulations (QAMs) are among the most used constellations, and are identified by the number of different symbols that can be transmitted.

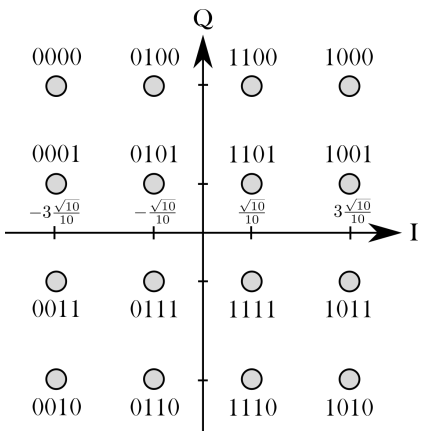


Figure 2.2: Constellation diagram corresponding to a 16-QAM constellation.

As an example, Fig. 2.2 shows the symbols and associated bits for a 2^4-QAM where the 4 bits are arranged according to a Gray labeling. After modulation, the symbols are converted into an electromagnetic wave that is transmitted through the channel.

In OFDM, the available bandwidth is divided into a set of N sub-band referred to as *subcarriers*. Orthogonality is achieved in the frequency domain by selecting a subcarrier spacing of Δ_f and applying a matching sinc-shaped pulse. The resulting spectrum is represented in Fig. 2.3, where it can be seen that each subcarrier is null at the frequencies corresponding to other subcarriers. In the time-domain, the duration of the corresponding signal is denoted by $T = \frac{1}{\Delta_f}$, and is referred to as the duration of an *OFDM symbol*. The entire time-frequency grid, formed by N subcarriers and M OFDM symbols, is referred to as the *resource grid (RG)*, while a single element in that grid is referred to as a *resource element (RE)* (Fig. 2.4).

Transmission over the RG is achieved by grouping the flow of symbols x to be transmitted into vectors of symbols $\mathbf{x}_m \in \mathcal{C}^N, m \in \{1, \cdots, M\}$ that are transmitted in parallel over all N subcarriers, effectively mapping each $x_{m,n} \in \mathcal{C}$ to the RE (m, n). To avoid any confusion between an OFDM symbol designating a column in the RG and a symbol that indicates a point in a constellation, the latter will also be referred to as a frequency baseband symbol (FBS), as the symbol mapping is carried out at the baseband over the available subcarriers.

2.1.2 The Wireless Channel

The channel model is the relation between the transmitted FBS x and the received FBS y, and can be expressed as

$$y = \mathrm{ch}(x), \tag{2.1}$$

where $\mathrm{ch}(\cdot)$ represents the distortions caused by the channel or the transceivers imperfections. In the following, we propose a derivation of the OFDM channel model inspired by [28].

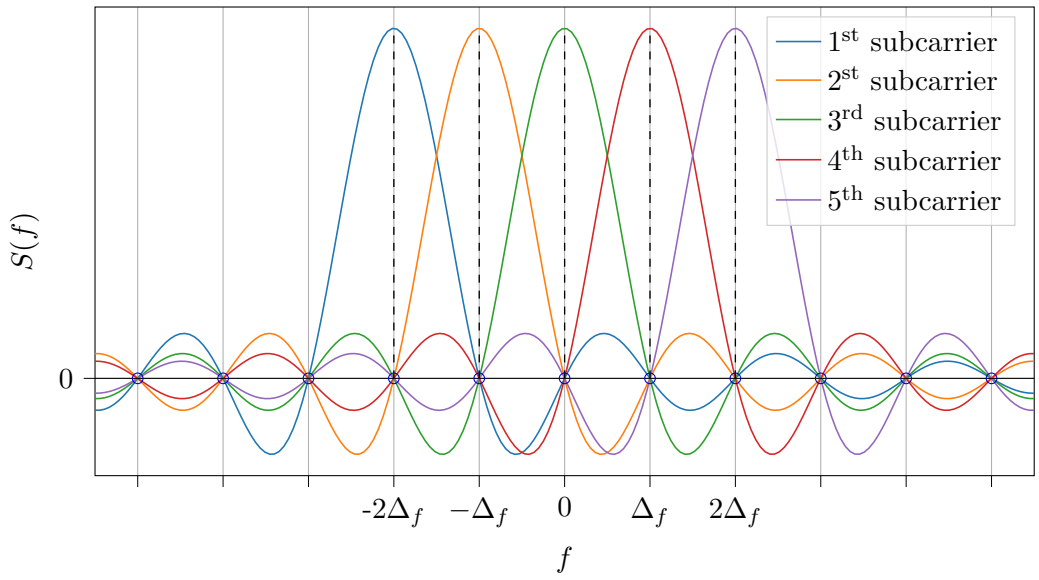


Figure 2.3: Representation of the amplitude of an OFDM spectrum $S(f)$ with $N = 5$ subcarriers centered around 0. Each subcarrier is null at the frequencies corresponding to other subcarriers, ensuring orthogonality.

Transmit filtering

The time-domain signal $s_m(t)$ corresponding to an OFDM symbol $\mathbf{x}_m$ is obtained by modulating each $x_{m,n}$ with a different transmit filer ϕ_n. During this modulation, a *cyclic prefix (CP)* is prepended to the symbol, and contains a copy of the last part of that symbol (Fig. 2.5). The length of the CP is denoted by T^{CP}, and total length is $T^{\text{tot}} = T^{\text{CP}} + T$.

If we denote by $\mathcal{N}$ the set of available subcarriers, the transmitted signal corresponding to the m^{th} OFDM symbol is

$$s_m(t) = \sum_{n \in \mathcal{N}} x_{m,n} \phi_n (t - mT^{\text{tot}}) \tag{2.2}$$

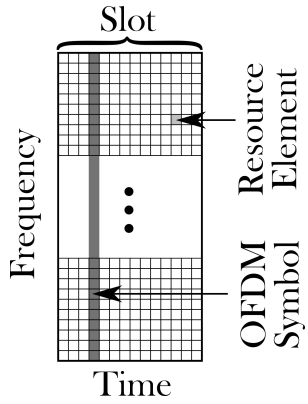


Figure 2.4: An OFDM resource grid.

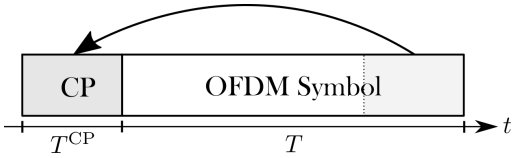


Figure 2.5: An OFDM symbol with its cyclic prefix appended.

where the filters ϕ_n are chosen such that $\phi_k(t) = \phi_k(t+T)$ when t is within the duration of the CP, i.e., when $t \in [0, T^{\mathrm{CP}}]$:

$$\phi_n(t) = \begin{cases} \frac{1}{\sqrt{T^{\mathrm{tot}}}} e^{j2\pi n \frac{t-T^{\mathrm{CP}}}{T}} & \text{if } t \in [0, T^{\mathrm{tot}}] \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

$$= \frac{1}{\sqrt{T^{\mathrm{tot}}}} \mathrm{rect}\left(\frac{t}{T^{\mathrm{tot}}} - \frac{1}{2}\right) e^{j2\pi n \frac{t-T^{\mathrm{CP}}}{T}}. \tag{2.4}$$

Note that we choose the transmit filters so that the average energy per OFDM symbol and per subcarrier is one. The transmitted signal corresponding to an OFDM slot is

$$s(t) = \sum_{m=0}^{M-1} s_m(t) = \sum_{m=0}^{M-1} \sum_{n \in \mathcal{N}} x_{m,n} \phi_n(t - mT^{\mathrm{tot}}). \tag{2.5}$$

Without CPs ($T^{\mathrm{CP}} = 0$), the spectrum of each filter $\phi_n(t)$ corresponds to a subcarrier as shown in Fig. 2.3. The removal of the CPs at the receiver-side therefore ensures the preservation of the orthogonality between subcarriers.

Channel

Let us denote by $g(\tau, t)$ the response of the channel at time t when excited with a impulse transmitted at time $t - \tau$. For a multipath channel, we have

$$g(\tau, t) = \sum_{p=0}^{P-1} a_p(t) \delta\left(\tau - \tau_p(t)\right) \tag{2.6}$$

where P is the number of different paths, $a_p(t)$ and $\tau_p(t)$ respectively denote the complex amplitude and time delay associated with the p^{th} path at time t, and $\delta(\cdot)$ is the Dirac function. The length of the CP is chosen to be at least equal to the longest delay, such as the impulse response of the channel is restricted to the interval $[0, T^{\mathrm{CP}}]$. The received signal can be expressed as

$$r(t) = \int_{-\infty}^{\infty} g(\tau, t) s(t - \tau) d\tau + \widetilde{n}(t) \tag{2.7}$$

$$= \int_{0}^{T^{\mathrm{CP}}} g(\tau, t) s(t - \tau) d\tau + \widetilde{n}(t) \tag{2.8}$$

where $\widetilde{n}(t)$ is a complex additive white Gaussian noise (AWGN) process with power spectral density (PSD) N_0 satisfying

$$\mathbb{E}\left[n(t)n^{\star}(t+\tau)\right] = N_0\delta(t-\tau). \tag{2.9}$$

As $a_p(t)$ and $\tau_p(t)$ typically vary slowly, it is common to assume that the channel is pseudo-stationnary, i.e., that it is constant over the duration of an OFDM symbol. On the m^{th} OFDM symbol, we therefore have $g(\tau,t) = g(\tau,t_m)$ with $t_m = mT^{\text{tot}}$, and (2.8) can be written as

$$r(t) = \int_0^{T^{\text{CP}}} g(\tau,t_m)s(t-\tau)d\tau + \widetilde{n}(t) \tag{2.10}$$

Receive filtering

At the receiver, the signal corresponding to the k^{th} subcarrier is obtained by filtering the received signal $r(t)$ with

$$\psi_k(t) = \begin{cases} \phi_k^{\star}(T^{\text{tot}}-t) & \text{if } t\in[0,T] \\ 0 & \text{otherwise} \end{cases} \tag{2.11}$$

$$= \text{rect}\left(\frac{t}{T}-\frac{1}{2}\right)\phi_k^{\star}(T^{\text{tot}}-t). \tag{2.12}$$

This receive filter, of duration T, is matched to the part of the transmit filter carrying the OFDM symbol, thus removing the CP. Since the channel impulse response was shorter than the CP, the filtered signal contains no intersymbol interference (ISI), and therefore we can focus on a single OFDM symbol. The output of the receive filter on the 0^{th} OFDM symbol and k^{th} subcarrier is

$$y_{0,k} = \int_{-\infty}^{\infty} r(t)\psi_k(T^{\text{tot}}-t)dt = \int_{T^{\text{CP}}}^{T^{\text{tot}}} r(t)\phi_k^{\star}(t)dt \tag{2.13}$$

$$= \int_{T^{\text{CP}}}^{T^{\text{tot}}}\left(\int_0^{T^{\text{CP}}} g(\tau,t_0)\left[\sum_{n\in\mathcal{N}} x_{0,n}\phi_n(t-\tau)\right]d\tau\right)\phi_k^{\star}(t)dt + \int_{T^{\text{CP}}}^{T^{\text{tot}}}\widetilde{n}(t)\phi_k^{\star}(t)dt \tag{2.14}$$

$$= \sum_{n\in\mathcal{N}} x_{0,n}\int_{T^{\text{CP}}}^{T^{\text{tot}}}\left(\int_0^{T^{\text{CP}}} g(\tau,t_0)\phi_n(t-\tau)d\tau\right)\phi_k^{\star}(t)dt + \int_{T^{\text{CP}}}^{T^{\text{tot}}}\widetilde{n}(t)\phi_k^{\star}(t)dt. \tag{2.15}$$

The inner integral can be expressed as

$$\int_0^{T^{\text{CP}}} g(\tau,t_0)\phi_n(t-\tau)d\tau = \int_0^{T^{\text{CP}}} g(\tau,t_0)\frac{1}{\sqrt{T^{\text{tot}}}}e^{j2\pi\frac{n}{T}t-\tau-T^{\text{CP}}}d\tau \tag{2.16}$$

$$= \frac{e^{j2\pi\frac{n}{T}\left(t-T^{\text{CP}}\right)}}{\sqrt{T^{\text{tot}}}}\underbrace{\int_0^{T^{\text{CP}}} g(\tau,t_0)e^{-j2\pi\frac{n}{T}\tau}d\tau}_{h_{0,n}}, \quad T^{\text{CP}} < t < T^{\text{tot}} \tag{2.17}$$

where h_n is the frequency response of the channel at the n^{th} subcarrier. The filtered signal (2.15) can now be written as

$$y_{0,k} = \sum_{n\in\mathcal{N}} x_{0,n} \int_{T^{\text{CP}}}^{T^{\text{tot}}} \frac{e^{j2\pi\frac{n}{T}\left(t-T^{\text{CP}}\right)}}{\sqrt{T^{\text{tot}}}} h_n \phi_k^\star(t)dt + \underbrace{\int_{T^{\text{CP}}}^{T^{\text{tot}}} \tilde{n}(t)\phi_k^\star(t)dt}_{n_{0,n}} \tag{2.18}$$

$$= \sum_{n\in\mathcal{N}} x_{0,n}h_{0,n} \int_{T^{\text{CP}}}^{T^{\text{tot}}} \phi_n(t)\phi_k^\star(t)dt + n_{0,n}. \tag{2.19}$$

The transmit filters are chosen to be orthogonal, i.e.,

$$\int_{T^{\text{CP}}}^{T^{\text{tot}}} \phi_n(t)\phi_k^\star(t) = \begin{cases} 1 & \text{if } k = n \\ 0 & \text{otherwise} \end{cases} \tag{2.20}$$

and therefore the FBS corresponding to the n^{th} subcarrier is

$$y_{0,n} = h_{0,n}x_{0,n} + n_{0,n} \tag{2.21}$$

where $n_{0,n}$ is an AWGN. The same derivation can be carried out for all subcarriers and OFDM symbols $m \in \{0, \cdots, M-1\}$, resulting in

$$\mathbf{y}_m = \mathbf{h}_m \odot \mathbf{x}_m + \mathbf{n}_m. \tag{2.22}$$

where the vectors $\mathbf{y}_m, \mathbf{h}_m$, $\mathbf{x}_m$, and $\mathbf{n}_m$ respectively contain the values of $y_{m,n}, h_{m,n}, x_{m,n}$, and $n_{m,n}$ for all subcarriers $n \in \mathcal{N}$.

A discrete-time model can be derived by replacing the transmit and receive filters by the inverse discrete Fourier transform (IDFT) and discrete Fourier transform (DFT) operators:

$$\mathbf{y}_m = \text{DFT}\left(\text{IDFT}\left(\mathbf{x}_m\right) \circledast \mathbf{g}_m + \widetilde{\mathbf{n}}_m\right) \tag{2.23}$$

$$= \text{DFT}\left(\text{IDFT}\left(\mathbf{x}_m\right) \circledast \mathbf{g}_m\right) + \mathbf{n}_m, \tag{2.24}$$

where the $\circledast$ operator denotes a cyclic convolution, $\mathbf{n}_m = \text{DFT}\left(\widetilde{\mathbf{n}}_m\right)$ is a vector of uncorrelated Gaussian noise, and $\mathbf{g}_m$ corresponds to the channel impulse response, i.e., $\mathbf{g}_m = \text{IDFT}(\mathbf{h}_m)$.

2.1.3 Receive Processing

The main benefit of OFDM systems is the simplified receiving process. If $h_{m,n}$ is known at the receiver, the transmitted symbols can be estimated from the received symbols:

$$\widehat{x}_{m,n} = \frac{y_{m,n}}{h_{m,n}} \tag{2.25}$$

$$= x_{m,n} + \underbrace{\frac{n_{m,n}}{h_{m,n}}}_{n'_{m,n}}. \tag{2.26}$$

This process is referred to as the *equalization*, and $n'_{m,n}$ is the post-equalization received noise with variance $\rho^2_{m,n}$. The next step is the *demapping*, in which the transmitted bits are estimated from $\hat{x}_{m,n}$. Hard demapping finds the symbol $\hat{\hat{x}}_{m,n} \in \mathcal{C}$ that is the closest to $\hat{x}_{m,n}$, i.e.,

$$\hat{\hat{x}}_{m,n} = \underset{c \in \mathcal{C}}{\operatorname{argmin}} |c - \hat{x}_{m,n}|^2 \tag{2.27}$$

and recovers the corresponding bits from the used constellation (as shown in Fig. 2.2 for 16-QAM). Soft demapping aims at providing probabilities over the transmitted bits in the form of log-likelihood ratios (LLRs). The LLR corresponding to the q^{th} bit on the RE (m, n) is given by

$$\begin{aligned} \text{LLR}_{m,n}(q) &= \ln\left(\frac{P(b_{m,n,q}=1)}{P(b_{m,n,q}=0)}\right) && (2.28)\\ &= \ln\left(\frac{\sum_{c\in\mathcal{C}_{q,1}} \exp\left(-\frac{1}{\rho^2_{m,n}}|\hat{x}_{m,n}-c|^2\right)}{\sum_{c\in\mathcal{C}_{q,0}} \exp\left(-\frac{1}{\rho^2_{m,n}}|\hat{x}_{m,n}-c|^2\right)}\right) && (2.29) \end{aligned}$$

where $\mathcal{C}_{q,0}$ ($\mathcal{C}_{b,1}$) is the subset of $\mathcal{C}$ which contains all symbols with the q^{th} bit set to 0 (1).

Note on channel estimation: Although is it sometimes assumed that the channel coefficients $h_{m,n}$ are known to the reciever, in practice only channel estimates $\hat{h}_{m,n}$ are available. Such estimations are usually obtained by transmitting pre-determined *pilot signals* $p_{m,n}$ on a set of fixed REs. The receiver can then estimate the channel on the REs (m, n) carrying pilots with $\hat{h}_{m,n} = \frac{y_{m,n}}{p_{m,n}}$, and extrapolate these channel estimates to the all remaining REs. To alleviate the overhead associated with pilot transmissions, end-to-end systems are able to perform pilotless communication by learning constellation that are not circularly symmetrical. More details on the channel estimation process will be presented in Chapter 4 and 5.

2.1.4 Uplink Multiple-Input Multiple-Output Systems

In multi-user multiple-input multiple-output (MU-MIMO) systems, K single antenna users communicate with a base station (BS) equipped with L antennas. In uplink transmissions, the vectors of transmitted and received symbols on each RE (m, n) are respectively denoted by $\mathbf{x}_{m,n} \in \mathcal{C}^K$ and $\mathbf{y}_{m,n} \in \mathbb{C}^L$. The channel model between all users and the BS antennas is

$$\mathbf{y}_{m,n} = \mathbf{H}_{m,n}\mathbf{x}_{m,n} + \mathbf{n}_{m,n} \tag{2.30}$$

where the noise vector is denoted by $\mathbf{n}_{m,n} \sim \mathcal{CN}\left(\mathbf{0}, \sigma^2\mathbf{I}_L\right)$ and $\mathbf{H}_{m,n} \in \mathbb{C}^{L\times K}$ is the matrix of channel coefficients. Although the transmit process is unchanged for each user, the detection of K users per RE by the BS requires a new equalization algorithm. Assuming that the channel matrix $\mathbf{H}_{m,n}$ is known, the optimal hard detection algorithm to minimize the probability of

symbol error is the maximum likelihood detector:

$$\hat{\hat{\mathbf{x}}}_{m,n} = \underset{\mathbf{x}_{m,n} \in \mathcal{C}^K}{\operatorname{argmin}} ||\mathbf{y}_{m,n} - \mathbf{H}_{m,n}\mathbf{x}_{m,n}||^2. \tag{2.31}$$

However, its complexity being exponential in K often prevents any practical implementations. A simple soft-detection algorithm is zero forcing, in which the constellation constraint on $\mathbf{x}_{m,n}$ is removed:

$$\begin{aligned}
\hat{\mathbf{x}}_{m,n} &= \underset{\mathbf{x}_{m,n} \in \mathbb{C}^K}{\operatorname{argmin}} ||\mathbf{y}_{m,n} - \mathbf{H}_{m,n}\mathbf{x}_{m,n}||^2 && (2.32)\\
&= \left(\mathbf{H}_{m,n}^{\mathsf{H}}\mathbf{H}_{m,n}\right)^{-1}\mathbf{H}_{m,n}^{\mathsf{H}}\mathbf{y}_{m,n} && (2.33)\\
&= \mathbf{x}_{m,n} + \left(\mathbf{H}_{m,n}^{\mathsf{H}}\mathbf{H}_{m,n}\right)^{-1}\mathbf{H}_{m,n}^{\mathsf{H}}\mathbf{n}_{m,n} && (2.34)
\end{aligned}$$

resulting in estimated symbols $\hat{\mathbf{x}}_{m,n}$ with zero intersymbol interferences. Note that the notations $\hat{\hat{\cdot}}$ and $\hat{\cdot}$ respectively denote the outputs obtained through hard and soft detection. The main drawback of this detector is that the noise can be significantly amplified on ill-conditionned channels, resulting in poor performances. To understand the cause of this effect, let us factorize the channel matrix $\mathbf{H}_{m,n}$ into its singular value decomposition (SVD) decomposition: $\mathbf{H}_{m,n} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^{\mathsf{H}}$, where $\mathbf{U} \in \mathbb{C}^{L\times L}$ and $\mathbf{V} \in \mathbb{C}^{K\times K}$ are unitary matrices, and $\mathbf{\Lambda} \in \mathbb{R}^{L\times K}$ is a rectangular diagonal matrix with element $[\lambda_1, \cdots, \lambda_K]$ on the diagonal. The symbol estimate can be re-written as

$$\hat{\mathbf{x}}_{m,n} = \mathbf{x}_{m,n} + \mathbf{V}\mathbf{\Lambda}^{-1}\mathbf{U}^{\mathsf{H}}\mathbf{n}_{m,n} \tag{2.35}$$

where $\mathbf{\Lambda}^{-1}$ has elements $[\frac{1}{\lambda_1}, \cdots, \frac{1}{\lambda_K}]$ on the diagonal. On ill-conditionned channels, some singular values $\lambda_l, l \in \{1, \cdots, L\}$ are very small, resulting in a matrix $\mathbf{\Lambda}^{-1}$ containing large entries that amplify the noise accordingly. To reduce the sensitivity of the detector to the sigular values of the channel, a regularization term $\sigma^2||\mathbf{x}_{m,n}||^2$ can be added to the objective function (2.32):

$$\hat{\mathbf{x}}_{m,n} = \underset{\mathbf{x}_{m,n} \in \mathbb{C}^K}{\operatorname{argmin}} ||\mathbf{y}_{m,n} - \mathbf{H}_{m,n}\mathbf{x}_{m,n}||^2 + \sigma^2||\mathbf{x}_{m,n}||^2. \tag{2.36}$$

The linear solution is equivalent to a Wiener filter and is referred to as the LMMSE detector. It will be discussed in more details in Chapters 3 and 5.

After equalization, the LLR on the transmitted bit q can be estimated for an OFDM symbol m, subcarrier n, and user k independently:

$$\mathrm{LLR}_{m,n,k}(q) = \ln\left(\frac{\sum_{c\in\mathcal{C}_{q,1}} \exp\left(-\frac{1}{\rho^2_{m,n,k}}\left|\hat{x}_{m,n,k} - c\right|^2\right)}{\sum_{c\in\mathcal{C}_{q,0}} \exp\left(-\frac{1}{\rho^2_{m,n,k}}\left|\hat{x}_{m,n,k} - c\right|^2\right)}\right) \tag{2.37}$$

where $\rho^2_{m,n,k}$ denotes the post-equalization noise variance corresponding to the user k on the RE (m, n). The transmission by K users of the bits corresponding an FBS (m, n) is depicted in Fig. 2.6, where the superscript (k) indicates that the quantity at hand is only considered for user k. Note that channel coding (decoding) blocs can be used at the transmitter (receiver) to detect and correct errors on the estimated bits.

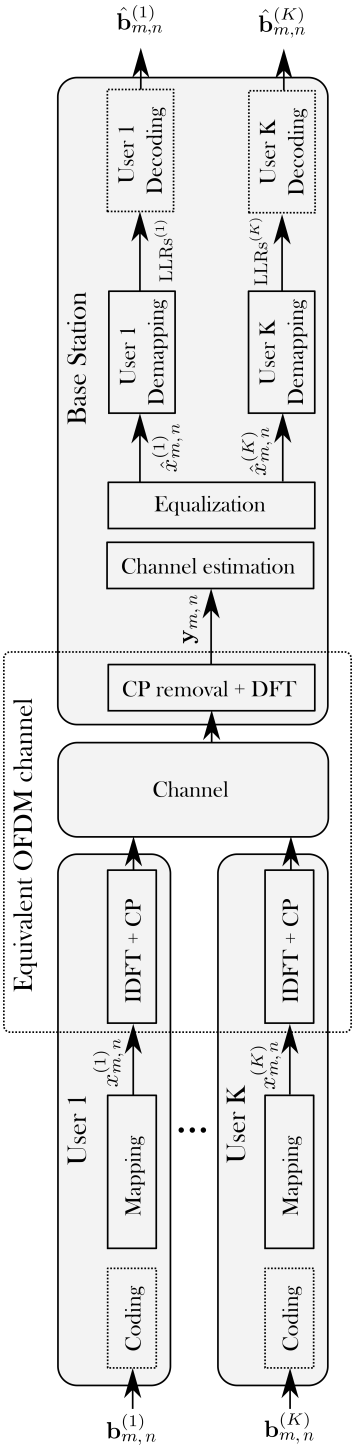


Figure 2.6: An uplink MU-MIMO system.

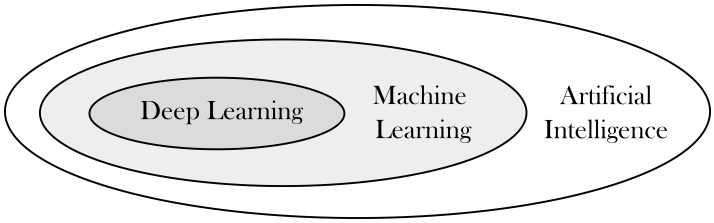


Figure 2.7: Deep Learning is a subset of Machine Learning, itself a subset of Artificial Intelligence.

2.2 Deep Learning for the Physical Layer

2.2.1 A General Introduction to Deep Learning

The concept of *artificial intelligence (AI)* can be traced back to 1842, when Ada Lovelace and other mathematicians started wondering whether machine could become intelligent. The notion of a computer "intelligence" is typically related to its ability to perform tasks commonly associated with intelligent beings. For example, one of the first successes of AI is the victory of IBM's Deep Blue computer against the chess world champion Garry Kasparov in 1997. The term of *machine learning (ML)* refers to AI systems capable of extracting patterns from raw data, and can be therefore considered as a subset of AI [29]. The performance of an ML-based algorithm typically depends on the quality of the information given, also known as the *features*. For example, whereas the height, age, or weight of a medical patient are numbers that represent relevant information, the pixel values of a scan are more difficult to interpret. A subset of ML algorithms therefore focuses on learning useful representations from raw input data. Among other solutions, *deep learning (DL)* systems use a suite of simple mathematical functions to learn as many intermediate representations of the input data. These simple functions are usually referred to as *layers*, and contain parameters that needs to be optimized. The relation between DL, ML, and AI is illustrated in Fig. 2.7.

The concept of gradient descent

The vast majority of DL-based systems use gradient-based algorithms to optimize the parameters of every layer. Such optimization problems involve a *loss function* that needs to be minimized, and that is related to the performance of the system. Let us define a simple loss function

$$l = L(x) = \frac{1}{2}x^2, \quad x \in \mathbb{R} \tag{2.38}$$

that needs to be minimized, i.e. we want to find the optimal x^* such that $x^* = \arg\min L(x)$. To that aim, we often use the derivative function, which gives the slope of the function $L(x)$

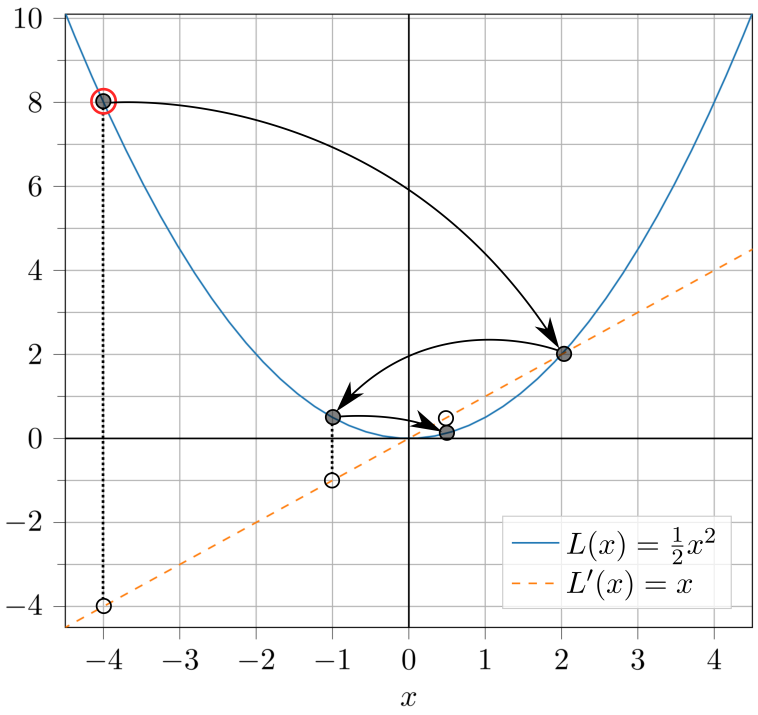


Figure 2.8: A visual representation of a gradient descent.

at any point x. The derivative of $L(x) = \frac{1}{2}x^2$ is simply given by

$$L'(x) = x. \tag{2.39}$$

$L(x)$ and $L'(x)$ are both represented in Fig. 2.8, where one can see that $L(x)$ is decreasing for negative values of x, and therefore the associated derivatives are negative. By opposition, $L(x)$ is increasing for positive values of x, and therefore the derivatives are positive.

It is clear that one should update x in the direction opposite to the derivative in order to minimize $L(x)$. From this observation, the following optimization step can be derived

$$x^{(i+1)} = x^{(i)} - \eta L'\left(x^{(i)}\right) \tag{2.40}$$

where the superscript (i) denote the i^{th} iteration of the algorithm and η is a hyperparameter that defines the size of the optimization step, also known as the *learning rate*. For the first iteration, $x^{(0)}$ is usually chosen randomly. To better grasp the intuition behind the algorithm, let us perform some iteration steps, starting with $x^{(0)} = -4$ and using $\eta = \frac{3}{2}$. These steps are illustrated in Fig. 2.8, where $x^{(0)}$ is represented by a red circle.

0. We begin with $x^{(0)} = -4$.

1. At the first iteration, we start by computing $L'\left(x^{(0)}\right) = L'(-4) = -4$. Then, one optimization step can be performed: $x^{(1)} = x^{(0)} - \frac{3}{2}L'\left(x^{(0)}\right) = -4 + \frac{3}{2} \times 4 = 2$.

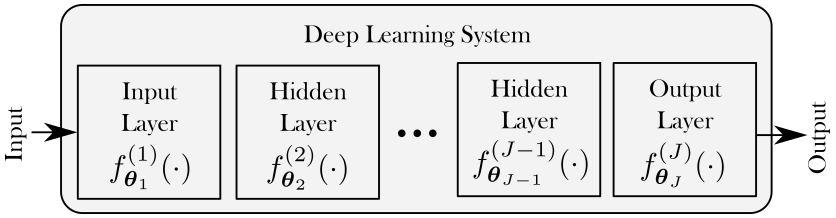


Figure 2.9: A DL system with J layers.

2. At the second iteration, we have $x^{(1)} = 2, L'(2) = 2$, and we can compute $x^{(2)} = x^{(1)} - \frac{3}{2}L'\left(x^{(1)}\right) = 2 - 3 = -1$.

3. At the third iteration, $x^{(2)} = -1, L'(-1) = -1$, and $x^{(3)} = x^{(2)} - \frac{3}{2}L'\left(x^{(2)}\right) = 0.5$.

Through these three iterations, $L(x)$ evolved from $L\left(x^{(0)}\right) = 8$ to $L\left(x^{(3)}\right) = 0.125$, thus becoming closer to the minimum value $L(x^*) = 0$ attainable with $x^* = 0$. Please note that the value $\eta = \frac{3}{2}$ is only used here to visualize the different optimization steps, as practical values are usually in the range $\left[10^{-5}, 10^{-2}\right]$. Finally, the derivative $L'(x)$, also denoted by $\frac{dl}{dx}$, can be extended to a *gradient* if multiple parameters are optimized. Such gradient is denoted by

$$\nabla_{\mathbf{x}} l = \left[\frac{\partial l}{\partial x_1}, \cdots, \frac{\partial l}{\partial x_K}\right]^{\mathsf{T}} \tag{2.41}$$

and is a vector containing the partial derivative of l with respect to (w.r.t.) the K parameters to be optimized $[x_1, \cdots, x_K]^{\mathsf{T}} = \mathbf{x}$. The algorithm performing (2.40), where x is updated in the direction opposite to the gradient, is therefore known as a *gradient descent* algorithm.

Gradient backpropagation

As defined previously, DL systems use a suite of simple mathematical functions, referred to as layers, to learn different representations of the input data. The first layer, to which the input data is fed, is the *input layer*, and the last layer, which outputs the estimated quantities, is the *output layer*. In between them, "deep" systems typically use multiple *hidden layers*. All these layers have trainable parameters that needs to be optimized so that the DL system can perform the desired task. Fig. 2.9 gives a visual representation of a DL system with J layers, each layer implementing a function $f^{(j)}_{\boldsymbol{\theta}_j}(\cdot)$ with trainable parameters $\boldsymbol{\theta}_j$.

For example, let us consider a DL system that learns to predict the time of flight t of a projectile thrown with an initial velocity v, and angle of launch α, and being thrown at a height h from the ground. For simplicity, the system is only composed of one input layer $y = f^{(1)}_{\boldsymbol{\theta}_1}(\mathbf{x})$ and one output layer $z = f^{(2)}_{\theta_2}(y)$, where $\boldsymbol{\theta}_1 = [\theta_{1,1}, \theta_{1,2}, \theta_{1,3}]^{\mathsf{T}} \in \mathbb{R}^3$ and $\theta_2 \in \mathbb{R}$ are the parameters that needs to be optimized. The system input is denoted by $\mathbf{x} = [v, \alpha, h]^{\mathsf{T}} \in \mathbb{R}^3$, and $y \in \mathbb{R}, z \in \mathbb{R}$ are the output of the first and second layers, respectively. The loss function calculates an error between the estimated time z and the true time of flight t and is defined by $L(z, t)$. Please note that we now aim at minimizing $L(z, t)$ w.r.t. the parameters $\boldsymbol{\theta}_1$ and θ_2,

in comparison with the minimization illustrated in Fig. 2.8 that was carried out w.r.t. x. The time of flight estimated by the NN is given by

$$z = f_{\theta_2}^{(2)}(y) = f_{\theta_2}^{(2)}\left(f_{\boldsymbol{\theta}_1}^{(1)}(\mathbf{x})\right) \tag{2.42}$$

and the associated loss is

$$l = L(z,t) = L\left(f_{\theta_2}^{(2)}\left(f_{\boldsymbol{\theta}_1}^{(1)}(\mathbf{x})\right), \frac{1}{2}mv^2\right). \tag{2.43}$$

We focus here on minimizing $L(z,t)$ w.r.t. the parameter of the input layer $\theta_{1,1}$, as the minimization w.r.t. other parameters would be similar. To apply the gradient descent algorithm on $\theta_{1,1}$, the chain rule of derivation is a useful tool:

$$\frac{\partial l}{\partial \theta_{1,1}} = \frac{\partial l}{\partial z}\frac{\partial z}{\partial \theta_{1,1}} = \frac{\partial l}{\partial z}\frac{\partial z}{\partial y}\frac{\partial y}{\partial \theta_{1,1}} \tag{2.44}$$

From here, it can be seen that to compute the derivative of the loss w.r.t. a parameter in the first layer, gradients on the loss function and on the second layer needs to be computed as well. In the following, we show that it is preferable to start by computing the gradient of the loss function $\frac{\partial l}{\partial z}$, then of the last layer $\frac{\partial z}{\partial y}$, and finally of the first layer $\frac{\partial y}{\partial \theta_{1,1}}$. This process, consisting in computing gradient from the loss function to the desired layer, is known as gradient *backpropagation*.

Performing backpropagation through a neural network

The core elements of *NNs* are *neurons*, each neuron j performing (Fig. 2.10)

$$o_j = \varphi(n_j) = \varphi\left(\sum_{k=1}^{K} \theta_{j,k} i_{j,k} + b_j\right) \tag{2.45}$$

where $i_{j,k} \in \mathbb{R}$ is the k^{th} input to the neuron (out of K), $o_j \in \mathbb{R}$ is the unique neuron output, $\varphi(\cdot)$ is an *activation function*, n_j is the neuron output prior to the activation function, $\theta_{j,k}$ is the trainable *weight* corresponding to the k^{th} input of the j^{th} neuron, and b_j is a trainable *bias*. In a neuron, the *trainable parameters* refer to the set comprising all the trainable weights and biases. For a neuron in the input layer, the vector $\mathbf{i}_j = [i_{j,1}, \cdots, i_{j,K}]^{\mathsf{T}}$ corresponds to the input vector $\mathbf{x}$, whereas $o_j = z$ for a neuron in the output layer (assuming a single output for simplicity). Without activation function, every neuron would perform a linear transformation, and an NN would simply be a composition of linear transformations, resulting in one large linear transformation. The aim of the activation function is to add non-linearities to the NN processing, and therefore to enable the handling of tasks that are more complex than linear regression problems.

Let us consider an NN with one input and one output layer, comprising one neuron each without bias ($b_1 = b_2 = 0$). The first layer takes an input vector of dimension three $\mathbf{x} \in \mathbb{R}^3$

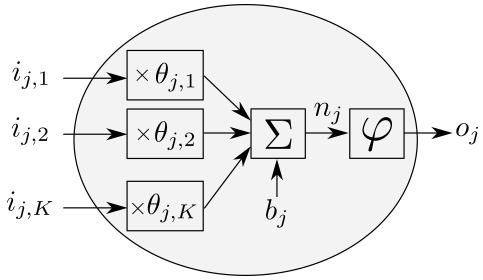


Figure 2.10: Representation of an artificial neuron.

and outputs a scalar y, while both the input y and the output z of the second layer are scalars. This NN can be expressed similarly to (2.42), with the first and second neurons being respectively implemented by $f^{(1)}_{\boldsymbol{\theta}_1}$ and $f^{(2)}_{\theta_2}$:

$$\begin{aligned} z &= f^{(2)}_{\theta_2}(y) \\ &= f^{(2)}_{\theta_2}\left(f^{(1)}_{\boldsymbol{\theta}_1}(\mathbf{x})\right) \\ &= \varphi\left(\theta_2\left(\varphi\left(\sum_{k=1}^{3}\theta_{1,k}x_k\right)\right)\right). \end{aligned} \tag{2.46}$$

In this example, the activation function is chosen to be the *sigmoid* function, defined as:

$$\varphi(a) = \frac{1}{1+\exp(-a)} \tag{2.47}$$

for which the derivative is

$$\frac{d\varphi(a)}{da} = \varphi(a)(1-\varphi(a)). \tag{2.48}$$

Finally, the loss in (2.43) is chosen to be the half of the squared error:

$$l = L(z,t) = \frac{1}{2}(z-t)^2 \tag{2.49}$$

for which the derivative w.r.t. z is

$$\frac{\partial l}{\partial z} = z - t. \tag{2.50}$$

First, let us compute the derivative of l w.r.t. to the parameter of the output layer θ_2:

$$\frac{\partial z}{\partial \theta_2} = \frac{\partial l}{\partial z}\frac{\partial z}{\partial \theta_2} = \frac{\partial l}{\partial z}\frac{\partial z}{\partial n_2}\frac{\partial n_2}{\partial \theta_2} \tag{2.51}$$

Each element of the right-hand side equation can be easily computed:

- $\frac{\partial l}{\partial z} = z - t$ as per (2.50).

- $\frac{\partial z}{\partial n_2} = \frac{\partial \varphi(n_2)}{\partial n_2} = \varphi(n_2)(1-\varphi(n_2)) = z(1-z)$.
- $\frac{\partial n_2}{\partial \theta_2} = \frac{\partial \theta_2 y}{\partial \theta_2} = y$

Substituting each element in (2.51), we obtain

$$\frac{\partial z}{\partial \theta_2} = (z-t)z(1-z)y = \delta_2 y \tag{2.52}$$

with

$$\delta_2 = \frac{\partial l}{\partial z}\frac{\partial z}{\partial n_2} = (z-t)z(1-z). \tag{2.53}$$

We can now compute the derivative of l w.r.t. the parameter of the first layer $\theta_{1,1}$:

$$\frac{\partial l}{\partial \theta_{1,1}} = \frac{\partial l}{\partial z}\frac{\partial z}{\partial n_2}\frac{\partial n_2}{\partial y}\frac{\partial y}{\partial n_1}\frac{\partial n_1}{\partial \theta_{1,1}} \tag{2.54}$$

- The first two elements have been computed above: $\frac{\partial l}{\partial z}\frac{\partial z}{\partial n_2} = \delta_2$
- $\frac{\partial n_2}{\partial y} = \frac{\partial \theta_2 y}{\partial y} = \theta_2$
- $\frac{\partial y}{\partial n_1} = \frac{\partial \varphi(n_1)}{\partial n_1} = y(1-y)$
- $\frac{\partial n_1}{\partial \theta_{1,1}} = \frac{\partial \theta_{1,1}x_1 + \theta_{1,2}x_2}{\partial \theta_{1,1}} = x_1$

which leads to

$$\frac{\partial l}{\partial \theta_{1,1}} = \delta_2 \theta_2 y(1-y)x_1 = \delta_1 x_1 \tag{2.55}$$

with

$$\delta_1 = \delta_2 \theta_2 y(1-y). \tag{2.56}$$

As one can see, the derivative needs to be computed starting from the last layer. For an NN with J layers, the backpropagation algorithm successively computes $\delta_J, \delta_{J-1}, \cdots, \delta_1$. Using the notation of (2.45), i.e., respectively denoting by o_j and $i_{j,k}$ the output and inputs of a neuron j and $\theta_{j,k}$ the parameter corresponding to the k^{th} input of this same neuron, the backpropagation to any layer j is given by:

$$\frac{\partial l}{\partial \theta_{j,k}} = i_{j,k}\delta_j, \text{ with } \delta_j = \begin{cases} (o_j - t)o_j(1-o_j) & \text{if j is an output neuron,} \\ \left(\sum_{m=1}^{M} \theta_{m,j}\delta_m\right) o_j(1-o_j) & \text{otherwise} \end{cases} \tag{2.57}$$

where M denotes the number of neurons in the $(j+1)^{\text{th}}$ layer, and the summation accounts for the fact that each layer can have multiple neurons, which was not considered in the example above for clarity. Please note that the expression (2.57) is only valid for an NN that

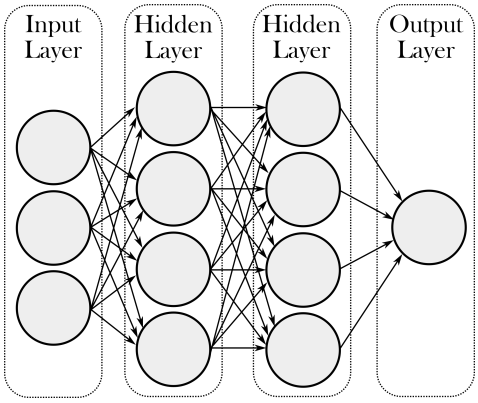


Figure 2.11: Representation of a neural network.

does not have any biases, uses logistic activation functions, and is evaluated using the loss $L(z,t) = \frac{1}{2}(z-t)^2$.

One of the most used type of NNs is the fully connected neural network (FCNN). In such NNs, each layer is *dense*, i.e., composed of neurons for which the inputs are the outputs of all the neurons from the preceding layer. A representation of an FCNN with 4 dense layers respectively containing three, four, four, and one neuron is presented in Fig. 2.11. Note that the NN output dimension corresponds to the number of neuron in the output layer, and can be greater than one.

The stochastic gradient descent algorithm

To train an NN such as the one defined in (2.46), a *dataset* comprising *features* and *labels* is needed. Let us denote by D_S the size of this dataset, which typically contains thousands or millions of such samples. In the example of a system that learns to predict the time of flight of a projectile, the features are vectors $\mathbf{x}^{[s]} = \left[v^{[s]}, \alpha^{[s]}, h^{[s]}\right]^\mathsf{T}, s \in \{1, \cdots, D_S\}$ and the labels are the associated measured time of flights $t^{[s]}, s \in \{1, \cdots, D_S\}$. If we denote by $\boldsymbol{\theta}$ the set containing all the trainable parameters and biases of the NN, a gradient descent iteration on the dataset is

$$\boldsymbol{\theta}^{(i+1)} = \boldsymbol{\theta}^{(i)} - \frac{\eta}{D_S} \sum_{s=1}^{D_S} \nabla_{\boldsymbol{\theta}} L(z^{[s]}, t^{[s]}) \tag{2.58}$$

where $z^{[s]}$ is the output of the NN corresponding to the input $\mathbf{x}^{[s]}$, and therefore depends on the parameters $\boldsymbol{\theta}$. However, such iteration would be of prohibitive complexity due to the large amount of samples in the dataset. The *stochastic gradient descent (SGD)* algorithm therefore randomly select *batches* of B_S samples from the dataset, and at each iteration performs a

gradient descent on a single batch only:

$$\boldsymbol{\theta}^{(i+1)} = \boldsymbol{\theta}^{(i)} - \frac{\eta}{B_S} \sum_{s=1}^{B_S} \nabla_{\boldsymbol{\theta}} L(z^{[s]}, t^{[s]}). \tag{2.59}$$

After each iteration, another random batch of samples is selected. The algorithm can be stopped by multiple factors, including when the loss reaches a given threshold or after a fixed number of iterations have been performed. Multiple enhancements of the SGD algorithm have been proposed, an example of which being the Adam optimizer [30] which sets individual learning rates for each parameter and computes a moving average of the gradient magnitude. A standard SGD algorithm that is stopped after I iteration is presented in algorithm 1.

Algorithm 1: A standard SGD algorithm

Initialize $\boldsymbol{\theta}^{(0)}$ randomly
for $i = 0, \cdots, I-1$ **do**
  ▷Select a random batch of B_S sample from the dataset
  ▷Compute one NN inference to obtain $z^{[s]}, s \in \{1, \cdots, B_S\}$
  ▷Evaluate the losses $L(z^{[s]}, t^{[s]}), s \in \{1, \cdots, B_S\}$
  ▷Perform one gradient descent iteration on the batch:
  $\boldsymbol{\theta}^{(i+1)} = \boldsymbol{\theta}^{(i)} - \frac{\eta}{B_S} \sum_{s=1}^{B_S} \nabla_{\boldsymbol{\theta}} L(z^{[s]}, t^{[s]})$
end

2.2.2 Layers and Activation Functions

To simplify the notations in section, we recall that the vector $\mathbf{m}_a$ and scalar $m_{a,b}$ formed by slicing the matrix $\mathbf{M}$ along its first and second dimensions can also be denoted by $[\mathbf{M}]_a$ and $[\mathbf{M}]_{a,b}$, respectively.

Dense layer

A dense layer, as presented previously, is composed of K neurons for which the inputs are the outputs of all J neurons from the preceding layer. If we respectively denote by $\mathbf{x} \in \mathbb{R}^J$ and $\mathbf{y} \in \mathbb{R}^K$ its input and output vectors of dimension J and K, a dense layer can be expressed by

$$\mathbf{y} = f(\mathbf{x}) = \varphi\left(\boldsymbol{\Theta}\mathbf{x} + \mathbf{b}\right) \tag{2.60}$$

where $\mathbf{b} \in \mathbb{R}^K$ and $\boldsymbol{\Theta} \in \mathbb{R}^{K \times J}$ and are respectively the vector of trainable biases and the matrix of trainable weights, and $\varphi(\cdot)$ is the activation function. One can see that each output $y_k = \varphi\left([\boldsymbol{\Theta}]_k^{\mathsf{T}}\, \mathbf{x} + b_k\right)$ corresponds to the processing done by the k^{th} neuron in the layer.

Convolutional layer

While dense layers are useful to process one-dimensional data, two- or three-dimensional inputs are usually processed by *convolutional* layers. Let us denote by $\mathbf{X} \in \mathbb{R}^{V \times H \times C}$ the 3D input of a convolutional layer. This could correspond to an image, where V and H are respectively the vertical and horizontal dimensions of the image, and the last dimension $C = 3$ corresponds to the red, green, and blue *channels*. Such convolutional layers have trainable *kernels* $\mathbf{K} \in \mathbb{R}^{K_V \times K_H \times C}$, with which the input is convoluted. (K_V, K_H) is referred to as the *kernel size*, and for simplicity let us assume that K_V and K_H are odd, i.e., $K_V = 2K'_V + 1$ and $K_H = 2K'_H + 1$. The output of the convolution at any position (x, y) is given by

$$[\text{conv}(\mathbf{X}, \mathbf{K})]_{x,y} = \sum_{v=0}^{K_V-1} \sum_{h=0}^{K_H-1} \sum_{c=0}^{C-1} [\mathbf{K}]_{v,h,c} [\mathbf{X}]_{x-K'_V+v, y-K'_H+h, c}. \tag{2.61}$$

For positions x (y) lower than K'_V (K'_H) or greater than $V - K'_V$ ($H - K'_H$), the indexes $x - K'_V + v$ ($y - K'_H + h$) are outside the dimension of $\mathbf{X}$. This problem can be dealt with by assuming that $[\mathbf{X}]_{x-K'_V+v, y-K'_H+h, c} = 0$ at these indexes, which corresponds to a convolution with zero-padding. Multiple kernels are usually defined for a given convolutional layer. Let us denote by F the number of kernels, also known as number of *filters*, and by $\mathbf{K}_f$ the f^{th} kernel, with $f \in \{0, \cdots, F-1\}$. The convolution with each filter defines a new output layer f, such as the convolution can be written as

$$[\text{conv}(\mathbf{X}, \mathbf{K})]_{x,y,f} = \sum_{v=0}^{K_V-1} \sum_{h=0}^{K_H-1} \sum_{c=0}^{C-1} [\mathbf{K}_f]_{v,h,c} [\mathbf{X}]_{x-K'_V+v, y-K'_H+h, c}. \tag{2.62}$$

Such a convolution is depicted in Fig. 2.12, where the input has $C = 3$ channels of dimension 10×10 (possibly corresponding to the RGB values of an image), the kernels have dimension $5 \times 5 \times 3$, and the first output layer (out of $F = 6$) is represented. To obtain the final outputs of the convolutional layer, biases $b_f \in \mathbb{R}$ are added for each convolution layer, and an activation function is applied:

$$[\mathbf{Y}]_{x,y,f} = \varphi\left([\text{conv}(\mathbf{X}, \mathbf{K})]_{x,y,f} + b_l\right) \tag{2.63}$$

$$= \varphi\left(\sum_{v=0}^{K_V-1} \sum_{h=0}^{K_H-1} \sum_{c=0}^{C-1} [\mathbf{K}_f]_{v,h,c} [\mathbf{X}]_{x-K'_V+v, y-K'_H+h, c} + b_f\right) \tag{2.64}$$

with $\mathbf{Y} \in \mathbb{R}^{V \times H \times F}$ being the output matrix.

The *receptive field* of an NN is defined as the dimension of the set of inputs that affects a single output. To increase the receptive field of a CNN, it is common to use *dilated* convolutions, in which the kernels are spread on the inputs:

$$[\text{dilated conv}_D(\mathbf{X}, \mathbf{K})]_{x,y,f} = \sum_{v=0}^{K_V-1} \sum_{h=0}^{K_H-1} \sum_{c=0}^{C-1} [\mathbf{K}_f]_{v,h,c} [\mathbf{X}]_{x-DK'_V+Dv, y-DK'_H+Dh, c} \tag{2.65}$$

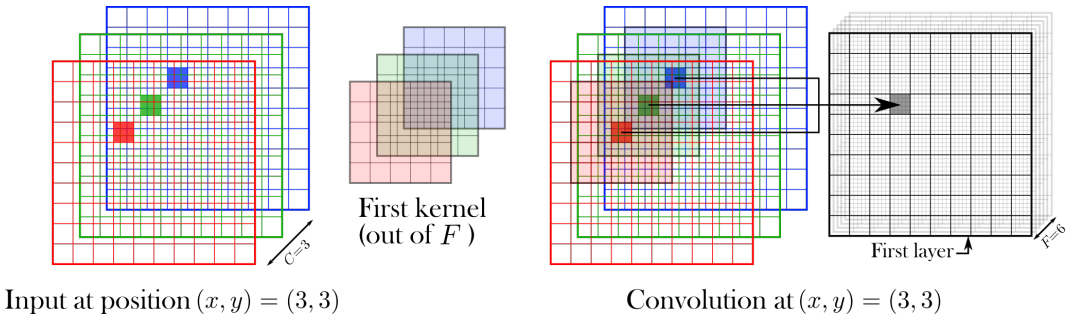


Figure 2.12: A convolution producing the first output layer (out of 6) at position $(x, y) = (3, 3)$.

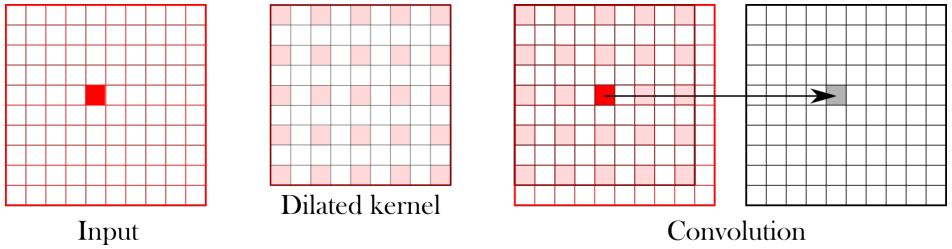


Figure 2.13: A dilated convolution at position $(x, y) = (5, 5)$, where $C = 1$, $F = 1$, $D = 2$.

where D is the dilation parameter. A dilated convolution with $D = 2$ is represented in Fig. 2.13.

Separable convolutional layer

A *separable* convolutional layer is composed of a depthwise convolution followed by a pointwise convolution. In depthwise convolutions, each kernel $K_f^{(\mathrm{d})} \in \mathbb{R}^{K_V \times K_H}$ has only two dimensions, and the number of kernels is the same as the number of input channels, i.e., $F = C$. Each kernel therefore acts separately on each input channel, resulting in

$$\mathbf{Y}^{(\mathrm{d})}_{x,y,c} = \sum_{v=0}^{K_V-1} \sum_{h=0}^{K_H-1} [\mathbf{K}_c^{(\mathrm{d})}]_{v,h} [\mathbf{X}]_{x-K'_V+v, y-K'_H+h, c} + b_c^{(\mathrm{d})} \tag{2.66}$$

where $\mathbf{Y}^{(\mathrm{d})}_{x,y,c} \in \mathbb{R}^{V \times H \times C}$ is the output matrix and $\mathbf{b}^{(\mathrm{d})} \in \mathbb{R}^C$ is the vector of trainable biases. Then, the pointwise convolution uses F kernels $\mathbf{K}_f^{(\mathrm{p})} \in \mathbb{R}^{1 \times 1 \times C}$:

$$\mathbf{Y}^{(\mathrm{p})}_{x,y,f} = \sum_{c=0}^{C-1} [\mathbf{K}_f^{(\mathrm{p})}]_{0,0,c} [\mathbf{Y}^{(\mathrm{d})}]_{x,y,c} + b_f^{(\mathrm{p})} \tag{2.67}$$

resulting in an output of dimension of $V \times H \times F$. Finally, the output of the separable convolution is given by

$$\mathbf{Y} = \varphi\left(\mathbf{Y}^{(\mathrm{p})}\right). \tag{2.68}$$

It has been shown that performing a separable convolution instead of a traditional convolution drastically reduces the number of computations and of trainable parameters while maintaining a similar level of performance [31].

Batch normalization layer

During training, the distribution of the outputs corresponding to each layer evolves as the trainable parameters are optimized. Therefore, each hidden layer needs to constantly readjust its parameters to follow the changes in its input distribution. This problem is amplified on deep NNs, as a change in the output distribution of the first layer can have a significant effect on the input distribution of the last layers. To tackle this problem, batch normalization layers normalize their inputs so that the corresponding distributions have optimized means and variances. Let us denote by $\mathbf{x}^{[s]} \in \mathbb{R}^J, s \in \{1, \cdots, B_S\}$ the input vectors of a batch normalization layer, where B_S is the batch size. The mean and variance of each element x_j can be estimated on the batch:

$$\mu_j = \frac{1}{B_S} \sum_{s=1}^{B_S} x_j^{[s]}, \text{ and } \sigma_j^2 = \frac{1}{B_S} \sum_{s=1}^{B_S} \left(x_j^{[s]} - \mu_j \right)^2. \tag{2.69}$$

Each dimension is then normalized separately to have zero-mean and unit variance:

$$\hat{x}_j^{[s]} = \frac{x_j^{[s]} - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}} \tag{2.70}$$

where ϵ is a small constant that is added to ensure numerical stability. The means and variances of each dimension j are then typically controlled by trainable parameters γ_j and β_j, respectively:

$$y_j^{[s]} = \gamma_j \hat{x}_j^{[s]} + \beta_j. \tag{2.71}$$

This reparametrization of the input distribution enables faster training and improves both the performance and the generalization properties of the NN [29].

Residual connections

NNs composed of many layers can be affected by the *vanishing gradient* problem, where the gradients that are backpropagated to the first layers become increasingly small, thus restraining the optimization of their parameters. Residual connections alleviate this effect by allowing the gradients to skip one or more layers. A simple implementation consists in adding the input of a (suite of) layer(s) to its output, as depicted in Fig. 2.14. NNs that contain residual connections are referred to residual networks, or *resnets*.

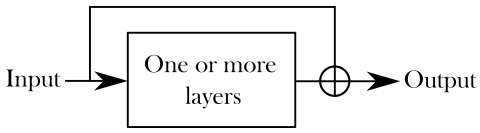


Figure 2.14: A residual connection.

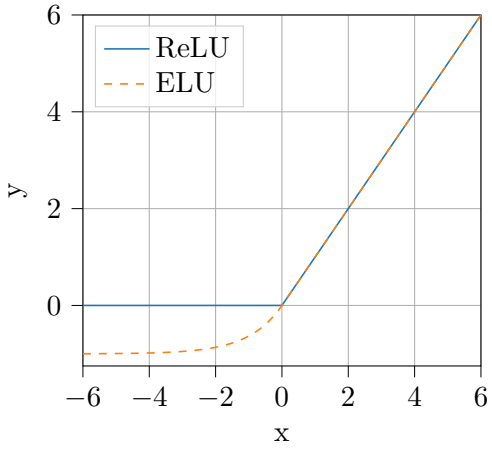


Figure 2.15: ReLU and ELU activation functions

The ReLU activation function

One of the most used activation function is the rectified linear unit (ReLU), depicted in Fig. 2.15:

$$y = \mathrm{ReLU}(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases} = \max(x, 0). \tag{2.72}$$

The main advantage of the ReLU is its simplicity, which allows for very efficient implementations. But NNs using this activation function might face the dying ReLU problem, where some neurons cannot output anything other than 0. Once a neuron is "dead", the gradient of its trainable parameters stays null as $\frac{d\,\mathrm{ReLU}(x)}{dx} = 0$ when $x < 0$, thus preventing further training.

The ELU activation function

To prevent the "dead" neuron problem, multiple variants of the ReLU have been proposed. One of them is the exponential linear unit (ELU) (Fig. 2.15), which mimics the ReLU function for positive x, but maintains a non-constant output for $x < 0$:

$$y = \mathrm{ELU}(x) = \begin{cases} \exp(x) - 1 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}. \tag{2.73}$$

ELU has been shown to outperform many other ReLU variants [32], but the use of the exponential function leads to longer computation times.

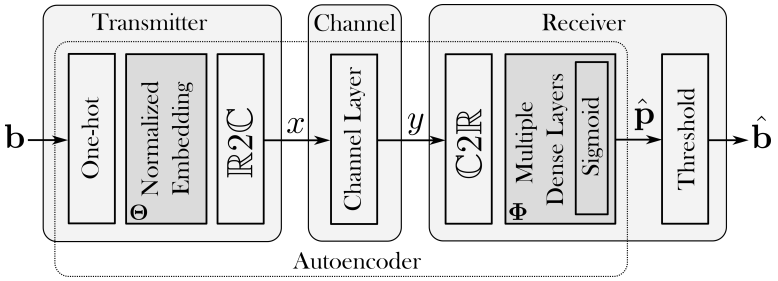


Figure 2.16: A communication system modeled as an autoencoder. The dark gray elements contain trainable parameters.

2.2.3 Optimizing Communication Systems through SGD

Modeling a communication system as an autoencoder

An autoencoder is a type of NN that aims to reconstruct its input at its output. Such NNs have one "bottleneck" layer of reduced dimensionality compared to its inputs and outputs, which means that an efficient, lower-dimensionality representation of the input data needs to be learned prior to this layer to enable a correct data reconstruction at the output. The first half of the NN is therefore usually referred to the encoder part, while the second half is the decoder part. A communication system can be seen as an autoencoder, as recovering the transmitted bits involve implementing a bit mapping at the transmitter (encoder) and demapping at the receiver (decoder) that is robust the channel distortions (bottleneck layer). A simple autoencoder-based communication system with simple AWGN channel is detailed in the following (Fig. 2.16).

At the transmitter, the first layer is a one-hot layer. This layer converts the vector of Q bits $\mathbf{b} \in \{0,1\}^Q$ into a vector of dimension 2^Q containing only zeros except a one at the position corresponding to the decimal representation of $\mathbf{b}$. For example, if $Q = 2$, $[0,0]^\mathsf{T}$ is converted to $[1,0,0,0]^\mathsf{T}$, $[0,1]^\mathsf{T}$ to $[0,1,0,0]^\mathsf{T}$, $[1,0]^\mathsf{T}$ to $[0,0,1,0]^\mathsf{T}$, and $[1,1]^\mathsf{T}$ is converted to $[0,0,0,1]^\mathsf{T}$. The next layer is a normalized embedding layer, which implements

$$f^{(1)}(\mathbf{a}) = \frac{\sqrt{2^Q}}{||\mathbf{\Theta}||_F}\mathbf{\Theta a} = \mathbf{W a} \tag{2.74}$$

where $\mathbf{\Theta} \in \mathbb{R}^{2\times 2^Q}$ is a matrix of trainable coefficients, and $\mathbf{W} \in \mathbb{R}^{2\times 2^Q}$ is normalized such that its column vectors have average energy of one. Finally, the $\mathbb{R}2\mathbb{C}$ layers converts 2 real numbers into a single complex number. The combination of the one-hot encoding, normalized embedding and $\mathbb{R}2\mathbb{C}$ layers outputs symbol x that have unit average energy, i.e., $\mathbb{E}\left[||x||_2^2\right] = 1$.

The AWGN channel is implemented as a channel layer that performs

$$y = x + n \tag{2.75}$$

where $n \sim \mathcal{CN}(0, \sigma^2)$ is a complex AWGN with variance σ^2. Such channel is differentiable

and therefore allows the gradient to be backpropagated from the receiver to the transmitter.

The first receiver layer is a $\mathbb{C}2\mathbb{R}$ layer, which converts a complex number into two real numbers. Then, the demapping is performed by multiple dense layers with trainable parameters denoted by $\mathbf{\Phi}$, the last layer being of dimension 2^Q and using the sigmoid activation function as defined in (2.47). The sigmoid has two advantages. First, its outputs are in the range $[0, 1]$, and thus can be interpreted as a probability vector $\hat{\mathbf{p}} = [\hat{p}_0, \cdots, \hat{p}_{Q-1}]^\mathsf{T}$, where each entry $\hat{p}_q$ corresponds to an estimated probability that a q^{th} bit equals one given y:

$$\hat{p}_q = \widehat{P}\left(b_q = 1|y\right). \tag{2.76}$$

Second, if we denote by *logits* the output of the last layers before the sigmoid activation function, we have

$$\widehat{P}\left(b_q = 1|y\right) = \frac{1}{1+\exp\left(-\text{logits}\right)} \iff \text{logits} = \ln\left(\frac{\widehat{P}\left(b_q = 1|y\right)}{\widehat{P}\left(b_q = 0|y\right)}\right). \tag{2.77}$$

These logits can thus be used as LLRs by a channel decoder, as illustrated in fig. 2.6.

For a perfect transmission, the vector of estimated probabilities $\hat{\mathbf{p}}$ matches the transmitted bit vector $\mathbf{b}$, i.e., $\hat{\mathbf{p}} = \mathbf{b}$, and therefore the communication system can truly be seen as an autoencoder. Finally, for each bit q, the threshold layer outputs

$$\hat{b}_q = \begin{cases} 1 & \text{if } \hat{p}_q > 0.5 \\ 0 & \text{otherwise} \end{cases}. \tag{2.78}$$

From NN-based systems to DL-enhanced systems

Estimating the bits that were transmitted is a binary classification problem, as for each bit the label is either 0 or 1. The loss function associated with such problem is the total binary cross-entropy, defined as

$$l = \mathbb{E}_y\left[L(\mathbf{b}, \hat{\mathbf{p}})\right] = \mathbb{E}_y\left[-\sum_{q=0}^{Q-1} b_q \cdot \log_2\left(\hat{p}_q\right) + \left(1-b_q\right) \cdot \log_2\left(1-\hat{p}_q\right)\right] \tag{2.79}$$

where the expected value reflects the fact that the metric should be independent of the noise realization n, and can be estimated through Monte-Carlo sampling with batches of size B_S:

$$l \approx \frac{1}{B_S}\sum_{s=1}^{B_S} L\left(\mathbf{b}^{[s]}, \hat{\mathbf{p}}^{[s]}\right). \tag{2.80}$$

Training such NN-based communication systems does not require any dataset, as two identical infinite sequences of bits can be obtained at the transmitter and at the receiver by initializing a random number generator with the same seed. These two sequences can be grouped into

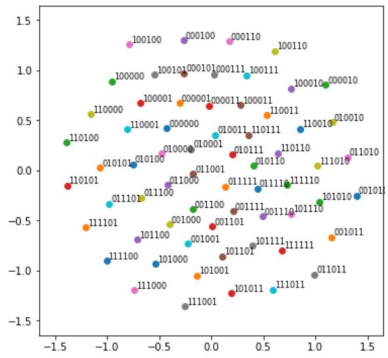


Figure 2.17: A learned constellation with $Q = 6$.

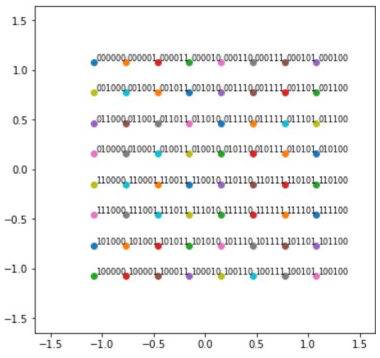


Figure 2.18: Constellation corresponding to a 64-QAM.

batches of bit vectors $\mathbf{b}$, enabling an SGD-based optimization:

$$\boldsymbol{\theta}^{(i+1)} = \boldsymbol{\theta}^{(i)} - \frac{\eta}{B_S} \sum_{s=1}^{B_S} \nabla_{\boldsymbol{\theta}} L\left(\mathbf{b}^{[s]}, \hat{\mathbf{p}}^{[s]}\right) \tag{2.81}$$

where $\boldsymbol{\theta}$ denotes the set of trainable parameters of the entire NN, i.e., $\boldsymbol{\theta} = \{\boldsymbol{\Theta}, \boldsymbol{\Phi}\}$. The constellation learned for a system trained with $Q = 6$ is depicted in Fig. 2.17 and can be compared to a 64-QAM modulation shown in Fig. 2.18. Performance evaluations will be carried out in Chapters 3, 4, and 5 for different DL-enhanced communication systems.

The differentiability of every layer in the communication system is key to achieve SGD-based optimization of every trainable parameters [1]. For example the loss function can not be applied to the estimated bits $\hat{\mathbf{b}}$, as the threshold layer (2.78) is not differentiable and therefore prevent the gradient to be backpropagated. Moreover, the trainable communication system presented in Fig. 2.16 is mostly composed of non-trainable layers, and the normalized embedding layer does not use any neurons. That explains why such systems are usually referred to as ML or DL-enhanced communication systems, instead of NN-based systems. Overall, this paradigm shift is increasingly visible, as the difference between training NNs to perform communication tasks and performing SGD on trainable communication systems has never been so thin.

An information theory perspective

Let us denote by $\mathcal{b}_q$ the random variable associated with the bit q. To simplify the notations, we denote by $P(b_q, y)$ and $\widehat{P}(b_q, y)$ the true and estimated probability that the q^{th} bit was transmitted, i.e.

$$P(b_q, y) = \begin{cases} P(\mathcal{b}_q = 1|y) & \text{if } b_q = 1 \\ P(\mathcal{b}_q = 0|y) & \text{if } b_q = 0 \end{cases} \tag{2.82}$$

[1] When no channel model is available, resulting in a non-differentiable channel layer, one can leverage deep reinforcement learning techniques to train the transmitter and the receiver in an alternating fashion [15], [33] or use generative adversarial networks to learn a channel model from available data [34], [35].

and

$$\widehat{P}(b_q, y) = \begin{cases} \widehat{P}(b_q = 1|y) & \text{if } b_q = 1 \\ \widehat{P}(b_q = 0|y) & \text{if } b_q = 0 \end{cases}. \tag{2.83}$$

The cross-entropy (CE) defined in (2.79) can be seen as an approximation through Monte Carlo sampling of the true CE, defined as

$$l = \sum_{q=0}^{Q-1} \mathbb{E}_y \left[H\left(P(b_q|y), \widehat{P}(b_q|y) \right) \right] \tag{2.84}$$

$$= -\sum_{q=0}^{Q-1} \int_y P(y) \sum_{b_q \in \{0,1\}} P(b_q|y) \log_2 \left(\widehat{P}(b_q|y) \right) dy \tag{2.85}$$

$$= -\sum_{q} \sum_{b_q} \int_y P(b_q, y) \log_2 \left(P(b_q) \frac{\widehat{P}(b_q|y) P(y)}{P(y) P(b_q)} \right) dy \tag{2.86}$$

$$= \underbrace{-\sum_{q=0}^{Q-1} \sum_{b_q} \int_y P(b_q, y) \log_2 \left(P(b_q) \right) dy}_{\sum_{q=0}^{Q-1} H(b_q)} - \sum_{q=0}^{Q-1} \sum_{b_q} \int_y P(b_q, y) \log_2 \left(\frac{\widehat{P}(b_q|y) P(y)}{P(y) P(b_q)} \right) dy \tag{2.87}$$

$$= \sum_{q=0}^{Q-1} H(b_q) - \sum_{q=0}^{Q-1} \sum_{b_q} \int_y P(b_q, y) \log_2 \left(\frac{P(b_q|y) P(y)}{P(y) P(b_q)} \frac{\widehat{P}(b_q|y)}{P(b_q|y)} \right) dy \tag{2.88}$$

$$= Q - \underbrace{\sum_{q=0}^{Q-1} \sum_{b_q} \int_y P(b_q, y) \log_2 \left(\frac{P(b_q, y)}{P(y) P(b_q)} \right) dy}_{\sum_{q=0}^{Q-1} I(b_q, y)} - \sum_{q=0}^{Q-1} \sum_{b_q} \int_y P(y) P(b_q|y) \log_2 \left(\frac{\widehat{P}(b_q|y)}{P(b_q|y)} \right) dy \tag{2.89}$$

$$= Q - \sum_{q=0}^{Q-1} I(b_q, y) + \underbrace{\sum_{q=0}^{Q-1} \int_y P(y) \sum_{b_q} P(b_q|y) \log_2 \left(\frac{P(b_q|y)}{\widehat{P}(b_q|y)} \right) dy}_{\sum_{q=0}^{Q-1} \mathbb{E}_y \left[D_{\mathrm{KL}} \left(P(b_q|y) || \widehat{P}(b_q|y) \right) \right]} \tag{2.90}$$

$$= Q - \underbrace{\left(\sum_{q=0}^{Q-1} I(b_q, y) - \sum_{q=0}^{Q-1} \mathbb{E}_y \left[D_{\mathrm{KL}} \left(P(b_q|y) || \widehat{P}(b_q|y) \right) \right] \right)}_{C}. \tag{2.91}$$

The first term of C is mutual information between all b_q and y, and corresponds to the maximum information rate that can be achieved assuming an ideal bit-metric decoding (BMD) receiver [36] This term both depends on the transmitter and on the channel model. The second term of C is the sum of the expected values of the KL-divergence between the true posterior probability $P(b_q|y)$ and the one estimated by the proposed receiver, and corresponds to a rate loss due to a suboptimal receiver. It can be seen as a measure of distance between

the probabilities that would be computed by an ideal receiver and the ones estimated by our NN-based implementation. Minimizing l therefore jointly optimizes the transmitter and the receiver to both maximize the information rate of the transmission and refine the estimated bit probabilities. Finally, C is an achievable rate assuming a mismatched BMD receiver [16] meaning that improvements in C directly translate to an improved BER performance.

— 3 —

HyperMIMO: a Deep HyperNetwork-Based MIMO Detector

3.1 Motivation

As introduced in Section 1.2, MU-MIMO is seen as a key technology to unlock the gains envisioned for beyond-5G systems. But optimal detection in such MIMO systems is known to be NP-hard [37], and less complex approaches usually suffer from unsatisfying performance on correlated channels or become impractical with large number of receive antennas. Examples of such approaches include the LMMSE detector [38], the approximate message passing (AMP) algorithm [39], and its extension to correlated channels [40]. Recently, advances in MIMO detection have been made using DL to improve the equalization block, detailed in Section 2.1, and which corresponds to a block-based optimization as depicted in Fig. 1.2a. One technique consists in using an NN to select a traditional detection algorithm from a predefined set [41]. According to available CSI, the algorithm with lowest complexity that enables a block error rate (BLER) lower than a predefined threshold is chosen. Another technique is to design an NN that directly performs the detection. One example is DetNet [42], which can be viewed as an unfolded recurrent neural network (RNN) where each iteration is made of three dense layers. Although it achieves encouraging results on Rayleigh channels, DetNet's performance on correlated channels is not satisfactory and it suffers from a prohibitive complexity. In [43], Mohammad et al. partially addressed this drawback by weights pruning. A third promising approach is known as *deep unfolding*, and consists in infusing existing iterative algorithms with DL components [44]. One possibility is to add trainable parameters to such algorithms

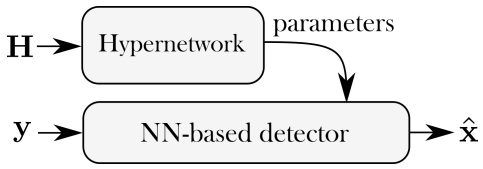


Figure 3.1: HyperMIMO: A hypernetwork generates the parameters of an NN-based detector.

and interpret the whole structure as an NN [45], [46], but most approaches still suffer from a performance drop on correlated channels. This was mitigated by the MMNet detector [47], which effectively achieves state-of-the-art performance on such channels. However, the need of retraining for each channel realization makes its practical implementation challenging.

In the following, we alleviate this issue using the emerging idea of *hypernetworks* [48], [49]. Applied to our setup, it consists in having a secondary NN, referred to as the hypernetwork, that generates for a given channel matrix an optimized set of weights for an NN-based detector. This scheme, which we referred to as *HyperMIMO* in our introductory paper [50], is illustrated in Fig. 3.1. Used with the MMNet detector from [47], HyperMIMO replaces the training procedure that would be required for each channel realization by a single inference of the hypernetwork. We have evaluated the proposed approach using simulations on spatially correlated channels. Our results show that HyperMIMO achieves performance close to that of MMNet trained for each channel realization, and outperforms the recently proposed OAMPNet [46]. They also reveal that HyperMIMO is robust to user mobility up to a certain point, which is encouraging for practical use. However, HyperMIMO still suffers from performance drops when evaluated with channels that vary significantly from the ones it has been trained with, and is only able to handle a fixed number of users. More recent works [51], [52] proposed to address these shortcomings and will be discussed in the closing section of this chapter.

Related literature

Although most of the published literature regarding NN-based block optimization focuses exclusively on the equalization step [45]–[47], [50]–[52], another line of research targets improved channel estimation. In [53], an NN architecture derived from a conventional LMMSE estimator is able to provide estimation gains on a wide range of channels with reduced complexity. Two CNN-based channel estimators are proposed in [54] for Massive MIMO mmWave transmissions, and either target lower computational complexity or reduced pilot overhead. Pilot overhead reduction is also studied in [55], where the pilot insertion and channel estimation blocks are jointly optimized to reduce the number of pilots required to achieve satisfactory channel estimation. However, these approaches require ground-truth of the channel realizations during training, which can only be approximated with costly measurement campaigns in practice. Finally, it has been proposed to improve the estimation of the bit probabilities by replacing the demapper with an NN, but this solution has only been studied for SISO setups [16], [56]–[58].

3.2 Framework

3.2.1 Problem Formulation and LMMSE Baseline

We consider a conventional MIMO uplink channel as presented in (2.30). In this section, a single RE (m, n) is considered, such that the channel transfer function can be simplified to

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n} \tag{3.1}$$

where $\mathbf{x} \in \mathcal{C}^K$ is the vector of transmitted symbols, $\mathbf{y} \in \mathbb{C}^L$ is the vector of received distorted symbols, $\mathbf{H} \in \mathbb{C}^{L\times K}$ is the channel matrix, and $\mathbf{n} \sim \mathcal{CN}(\mathbf{0}, \sigma^2\mathbf{I}_L)$ is the independent and identically distributed (i.i.d.) complex Gaussian noise with power σ^2 in each complex dimension. It is assumed that $\mathbf{H}$ and σ are perfectly known to the receiver. In the following, the problem of hard symbol detection is considered, in which the estimated symbol $\hat{\hat{\mathbf{x}}}$ must belong to the used constellation, i.e., $\hat{\hat{\mathbf{x}}} \in \mathcal{C}^K$.

The optimal maximum likelihood detector, as defined in (2.31), is known to be too complex for any practical implementation [37]. On the other hand, the zero-forcing equalizer (2.33) is known to perform poorly on ill-conditionned channels. To tackle both issues, one well-known scheme is the LMMSE estimator which aims to minimize the mean squared error (MSE)

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}' \in \mathbb{C}^K; \mathbf{x}' = \mathbf{W}\mathbf{y}} \mathbb{E}_{\mathbf{x},\mathbf{n}} \left[||\mathbf{x} - \mathbf{x}'||_2^2 \right] \tag{3.2}$$

by restricting to linear estimators, i.e., by left-multiplying $\mathbf{y}$ with a matrix $\mathbf{W} \in \mathbb{C}^{K\times L}$. A derivation can be obtained by finding the matrix $\mathbf{W}$ that nulls the gradient

$$\nabla_{\mathbf{W}} \mathbb{E}_{\mathbf{x},\mathbf{n}} \left[||\mathbf{x} - \mathbf{W}\mathbf{y}||_2^2 \right] = \mathbb{E}_{\mathbf{x},\mathbf{n}} \left[-2\mathbf{x}\mathbf{y}^{\mathsf{H}} + 2\mathbf{W}\mathbf{y}\mathbf{y}^{\mathsf{H}} \right] \stackrel{!}{=} 0 \tag{3.3}$$

$$\iff \mathbf{W} = \mathbb{E}_{\mathbf{x},\mathbf{n}} \left[\mathbf{x}\mathbf{y}^{\mathsf{H}} \right] \left(\mathbb{E}_{\mathbf{x},\mathbf{n}} \left[\mathbf{y}\mathbf{y}^{\mathsf{H}} \right] \right)^{-1} \tag{3.4}$$

$$\iff \mathbf{W} = \mathbf{H}^{\mathsf{H}} \left(\mathbf{H}\mathbf{H}^{\mathsf{H}} + \sigma^2 \mathbf{I}_K \right)^{-1}. \tag{3.5}$$

This allows for a closed-form expression of the solution to (3.2)

$$\hat{\mathbf{x}} = (\mathbf{H}^H\mathbf{H} + \sigma^2\mathbf{I}_K)^{-1}\mathbf{H}^H\mathbf{y}. \tag{3.6}$$

Because the transmitted symbols are known to belong to the finite alphabet $\mathcal{C}$, the closest symbol is typically selected for each user:

$$\hat{\hat{x}}_k = \arg\min_{x\in\mathcal{C}} ||\hat{x}_k - x||_2^2, \qquad \forall k \in \{1, \cdots, K\}. \tag{3.7}$$

Although sub-optimal, this approach has the benefit of being computationally tractable. Multiple schemes have been proposed to achieve a better performance-complexity trade-off among which DL-based algorithms form a particularly promising lead.

3.2.2 Deep Learning-based MIMO Detectors

As discussed previously, an interesting approach to design enhanced detectors is to add trainable parameters to existing schemes, and is often referred to as deep unfolding [44]. Traditional iterative algorithms are particularly suitable since they can be viewed as NN once unfolded. Typically, each iteration aims to further reduce the MSE and comprises a linear step followed by a non-linear denoising step. The estimate $\hat{\mathbf{x}}^{(i+1)}$ at the $(i+1)^{\text{th}}$ iteration is

$$\begin{aligned} \boldsymbol{\kappa}^{(i)} &= \hat{\mathbf{x}}^{(i)} + \mathbf{A}^{(i)} \left(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}^{(i)} + \mathbf{c}^{(i)} \right) \\ \hat{\mathbf{x}}^{(i+1)} &= \chi^{(i)} \left(\boldsymbol{\kappa}^{(i)}, \tau^{(i)} \right) \end{aligned} \tag{3.8}$$

where the superscript (i) is used to refer to the i^{th} iteration and $\hat{\mathbf{x}}^{(0)}$ is set to $\mathbf{0}$. $\tau^{(i)}$ denotes the estimated variance of the components of the noise vector $\boldsymbol{\kappa}^{(i)} - \mathbf{x}^{(i)}$ at the input of the denoiser, which is assumed to be i.i.d.. Iterative algorithms differ by their choices of matrices $\mathbf{A}^{(i)} \in \mathbb{C}^{K \times L}$, bias vectors $\mathbf{c}^{(i)} \in \mathbb{C}^{K}$, and denoising functions $\chi^{(i)}(\cdot)$. A limitation of most detection schemes is their poor performance on correlated channels. OAMP [40] mitigates this issue by constraining both the linear step and the denoiser. OAMPNet [46] improves the performance of OAMP by adding two trainable parameters per iteration, which respectively scales the matrix $\mathbf{A}^{(i)}$ and the channel noise variance σ^2. MMNet [47] goes one step further by making all matrices $\mathbf{A}^{(i)}$ trainable and by relaxing the constraint on $\boldsymbol{\kappa}^{(i)} - \mathbf{x}^{(i)}$ being identically distributed. Although MMNet achieves state-of-the-art performance on spatially-correlated channels, it needs to be re-trained for each channel matrix, which makes it unpractical.

3.2.3 Hypernetworks

Hypernetworks were introduced in [59] as NNs that generate the parameters of other NNs. The concept was first used in [48] in the context of image recognition. The goal was to predict the parameters of an NN given a new sample so that it could recognize other objects of the same class without the need for training. This same idea was also leveraged to generate images of talking heads [49]. In this later work, a picture of a person is fed to a hypernetwork that computes the weights of a second NN. This second NN then generates realistic images of the same person with different facial expressions. Motivated by these achievements, we propose to alleviate the need of MMNet to be retrained for each channel realization using hypernetworks.

3.3 HyperMIMO

The key idea of our approach is to replace the training process required by MMNet for each channel realization by a single inference through a trained hypernetwork. We first present a variation of MMNet which reduces its number of parameters, and then introduce the architecture of the hypernetwork, where a relaxed form of weight sharing is used to decrease its output dimension. Both reducing the number of parameters of MMNet and using weight sharing in the hypernetwork are crucial to obtain a system of reasonable complexity. The combination of the hypernetwork together with MMNet form the HyperMIMO system, schematically shown in Fig. 3.1.

3.3.1 MMNet with Less Parameters

To reduce the number of parameters of MMNet, we perform the QR-decomposition of the channel matrix, $\mathbf{H} = \mathbf{QR}$, where $\mathbf{Q}$ is an $L \times L$ orthogonal matrix and $\mathbf{R}$ an $L \times K$ upper triangular matrix. It is assumed that $L > K$, and therefore $\mathbf{R} = \begin{bmatrix} \mathbf{R_A} \\ \mathbf{0} \end{bmatrix}$ where $\mathbf{R_A}$ is of size $K \times K$, and $\mathbf{Q} = [\mathbf{Q_A} \boldsymbol{Q_B}]$ where $\mathbf{Q_A}$ has size $L \times K$. We define $\bar{\mathbf{y}} \coloneqq {\mathbf{Q_A}}^H \mathbf{y}$ and $\bar{\mathbf{n}} \coloneqq {\mathbf{Q_A}}^H \mathbf{n}$, and rewrite (3.1) as

$$\bar{\mathbf{y}} = \mathbf{R_A}\mathbf{x} + \bar{\mathbf{n}}. \tag{3.9}$$

Note that $\bar{\mathbf{n}} \sim \mathcal{CN}(\mathbf{0}, \sigma^2 \mathbf{I}_K)$. MMNet sets $\mathbf{c}^{(i)}$ to $\mathbf{0}$ for all i and uses the same denoiser for all iterations, which are defined by

$$\begin{aligned} \boldsymbol{\kappa}^{(i)} &= \hat{\mathbf{x}}^{(i)} + \boldsymbol{\Theta}^{(i)} \left(\bar{\mathbf{y}} - \mathbf{R_A} \hat{\mathbf{x}}^{(i)} \right) \\ \hat{\mathbf{x}}^{(i+1)} &= \chi \left(\boldsymbol{\kappa}^{(i)}, \boldsymbol{\tau}^{(i)} \right) \end{aligned} \tag{3.10}$$

where $\boldsymbol{\Theta}^{(i)}$ is a $K \times K$ complex matrix whose components need to be optimized for each channel realization. The main benefit of leveraging the QR-decomposition is that the dimension of the matrices $\boldsymbol{\Theta}^{(i)}$ to be optimized is $K \times K$ instead of $K \times L$, which is the dimension of $\mathbf{A}^{(i)}$ in (3.8). This is significant since the number of active users K is typically much smaller than the number of antennas L of the BS.

The noise at the input of the denoiser $\boldsymbol{\kappa}^{(i)} - \mathbf{x}^{(i)}$ is assumed to be independent but not identically distributed in MMNet. The vector of estimated variances at the i^{th} iteration is denoted by $\boldsymbol{\tau}^{(i)} \in \mathbb{R}^K$ and computed by

$$\boldsymbol{\tau}^{(i)} = \frac{\boldsymbol{\psi}^{(i)}}{K} \left(\frac{||\mathbf{I}_K - \boldsymbol{\Theta}^{(i)} \mathbf{R_A}||_F^2}{||\mathbf{R_A}||_F^2} \left[||\bar{\mathbf{y}} - \mathbf{R_A} \hat{\mathbf{x}}^{(i)}||_2^2 - L\sigma^2 \right]^+ + ||\boldsymbol{\Theta}^{(i)}||_F^2 \sigma^2 \right) \tag{3.11}$$

where $[x]^+ = \max(0, x)$, and $\boldsymbol{\psi}^{(i)} \in \mathbb{R}^K$ needs to be optimized for each channel realization. Further details on the origin of this equation can be found in [40]. The denoising function in MMNet is the same for all iterations, and is chosen to minimize the MSE $\mathbb{E}_{\mathbf{x}} \left[||\hat{\mathbf{x}}^{(I)} - \mathbf{x}||_2^2 \right]$

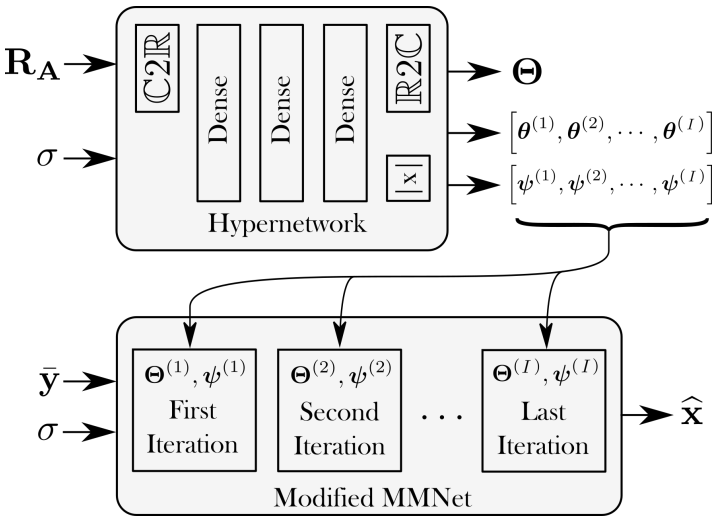


Figure 3.2: Detailed architecture of HyperMIMO

assuming the noise is independent and Gaussian distributed. This is achieved by applying element-wisely to $(\boldsymbol{\kappa}^{(i)}, \boldsymbol{\tau}^{(i)})$

$$\chi(\kappa, \tau) = \frac{1}{\sum_{x \in \mathcal{C}} \exp\left(-\frac{|\kappa - x|^2}{\tau}\right)} \sum_{x \in \mathcal{C}} x \exp\left(-\frac{|\kappa - x|^2}{\tau}\right). \tag{3.12}$$

MMNet consists of I layers performing (3.10), and a hard decision as in (3.7) to predict the final estimate $\hat{\hat{\mathbf{x}}}$. One could also use $\hat{\mathbf{x}}^{(I)}$ to predict bit-wise LLRs.

3.3.2 HyperMIMO Architecture

Fig. 3.2 shows in details the architecture of HyperMIMO. As our variant of MMNet operates on $\bar{\mathbf{y}}$, the hypernetwork is fed with $\mathbf{R_A}$ and the channel noise standard deviation σ. Note that because $\mathbf{R_A}$ is upper triangular, only $K(K+1)/2$ non-zero elements need to be fed to the hypernetwork. Moreover, using this matrix as input instead of $\mathbf{H}$ has been to found to be critical to achieve high performance. As detailed previously, the number of parameters that need to be optimized in MMNet was reduced by leveraging the QR-decomposition. To further decrease the number of outputs of the hypernetwork, we adopt a relaxed form of weight sharing inspired by [48]. Instead of computing the elements of each $\boldsymbol{\Theta}^{(i)}, i = \{1, \ldots, I\}$, the hypernetwork outputs a single matrix $\boldsymbol{\Theta}$ as well as I vectors $\boldsymbol{\theta}^{(i)} \in \mathbb{R}^K$. For each iteration i, $\boldsymbol{\Theta}^{(i)}$ is computed by

$$\boldsymbol{\Theta}^{(i)} = \boldsymbol{\Theta}\left(\mathbf{I}_K + \operatorname{diag}\left(\boldsymbol{\theta}^{(i)}\right)\right). \tag{3.13}$$

The idea is that all matrices $\boldsymbol{\Theta}^{(i)}$ differ by a per-column scaling different for each iteration. We have experimentally observed that scaling of the rows leads to worse performance.

Because $\mathbf{R_A}$ is complex-valued, a $\mathbb{C}2\mathbb{R}$ layer maps the complex elements of $\mathbf{R_A}$ to real ones, by concatenating the real and imaginary parts of the complex scalar elements. To generate a

complex-valued matrix $\mathbf{\Theta}$, a $\mathbb{R}2\mathbb{C}$ layer does the reverse operation of $\mathbb{C}2\mathbb{R}$. The hypernetwork also needs to compute the values of the I vectors $\boldsymbol{\psi}^{(i)}$. Because the elements of these vectors must be positive, an absolute-value activation function is used in the last layer.

HyperMIMO, which comprises the hypernetwork and MMNet, is trained by minimizing the MSE between the transmitted and estimated symbols, denoted by $\widehat{\mathbf{x}} = \widehat{\mathbf{x}}^{(I)}$:

$$\mathcal{L} = \mathbb{E}_{\mathbf{x},\mathbf{H},\mathbf{n}} \left[||\widehat{\mathbf{x}} - \mathbf{x}||_2^2 \right] \tag{3.14}$$

Finally, the expected value can be approximated through Monte-Carlo sampling, by sending batches of B_S samples:

$$\mathcal{L} \approx \frac{1}{B_S} \sum_{s=1}^{B_S} \left\| \widehat{\mathbf{x}}^{[s]} - \mathbf{x}^{[s]} \right\|_2^2 \tag{3.15}$$

Note that this loss differs from the one of [47], which is $\frac{1}{I} \sum_{i=1}^{I} \mathbb{E}_{\mathbf{x},\mathbf{H},\mathbf{n}} \left[||\widehat{\mathbf{x}}^{(i)} - \mathbf{x}||_2^2 \right]$. When training HyperMIMO, the hypernetwork and MMNet form a single NN, such that the output of the hypernetwork are the weights of MMNet. The only trainable parameters are therefore the ones of the hypernetwork. When performing gradient descent, their gradients are backpropagated through the parameters of MMNet.

3.4 Experiments

HyperMIMO was evaluated by simulations. This section starts by introducing the considered spatially correlated channel model. Next, details on the simulation setting and training process are provided. Finally, the obtained results are presented and discussed.

3.4.1 Channel Model

The local scattering model with spatial correlation presented in [26, Ch. 2.6] and illustrated in Fig. 3.3 is considered. The BS is assumed to be equipped with a uniform linear array of L antennas, located at the center of a 120°-cell sector in which K single-antenna users are dropped with random nominal angles $\varphi_k,\ k \in \{1, \cdots, K\}$. Perfect power allocation is assumed, leading to all users appearing to be at the same distance r from the BS and an average gain of one. The BS is assumed to be elevated enough to have no scatterers in its near field, such that the scattering is only located around the users. Given a user k, the multipath components reach the BS with normally distributed angles with mean φ_k and variance σ_φ^2. For small enough σ_φ, a valid approximation of the channel covariance matrix is $\mathbf{C}_k \in \mathbb{C}^{L \times L}$ with components

$$[C_k]_{a,b} = e^{2\pi j d(a-b)\sin(\varphi_k)} e^{-\frac{\sigma_\varphi^2}{2}(2\pi d(a-b)\cos(\varphi_k))^2} \tag{3.16}$$

where d is the antenna spacing measured in multiples of the wavelength. For a given user k, a random channel vector $\mathbf{h}_k \sim \mathcal{CN}(\mathbf{0}, \mathbf{C}_k)$ is sampled by computing

$$\mathbf{h}_k = \mathbf{L}_k \mathbf{\Lambda}_k^{\frac{1}{2}} \mathbf{L}_k^H \mathbf{e} \tag{3.17}$$

where $\mathbf{e}$ is sampled from $\mathcal{CN}(\mathbf{0}, \mathbf{I}_L)$ and $\mathbf{L}_k \mathbf{\Lambda}_k \mathbf{L}_k^H$ is the eigenvalue decomposition of $\mathbf{C}_k$. The signal-to-noise ratio (SNR) of the transmission is defined by

$$\text{SNR} = \frac{\mathbb{E}\left[\frac{1}{N_r}\|\mathbf{y}\|_2^2\right]}{\sigma^2} = \frac{1}{\sigma^2}. \tag{3.18}$$

3.4.2 Simulation Setting

The number of antennas that equip the BS was set to $L = 12$, and the number of users to $K = 6$. Quadrature phase-shift keying (QPSK) modulation was used. The standard deviation of the multipath angle distribution σ_φ was set to 10°, which results in highly correlated channel matrices. The number of layers of MMNet in the HyperMIMO detector was set to $I = 5$. The hypernetwork was made of 3 dense layers (see Fig. 3.2). The first layer had a number of units matching the number of inputs, the second layer 75 units, and the last layer a number of units matching the number of parameters of the detector. The first two dense layers used ELU activation functions, and the last dense layer had no activation functions.

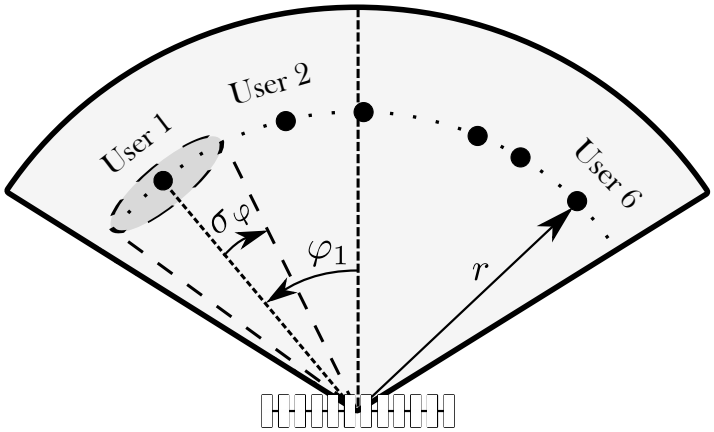


Figure 3.3: Considered channel model. The BS has no scatters in its near field, and scattering is only located near users.

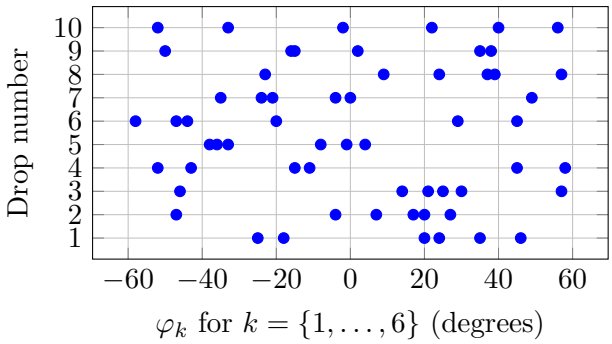


Figure 3.4: Ten randomly generated user drops

Our simulations revealed that training with randomly sampled user drops leads to suboptimal results. Therefore, HyperMIMO was trained with fixed channel statistics, i.e., fixed user positions. If this might seem unpromising, our results show that HyperMIMO is still robust to user mobility (see Section 3.4.3). Moreover, our scheme only has 10× more parameters than MMNet as proposed in [47], which allows it to be quickly re-trained in the background when the channel statistics change significantly. Note that this is different from MMNet that needs to be retrained every time the channel matrix changes, which is considerably more computationally demanding. Moreover, it is possible that further investigations on the hypernetwork architecture may alleviate this issue.

Given a user drop, HyperMIMO was trained by randomly sampling channel matrices $\mathbf{H}$, SNRs from the range [0,10]dB, and symbols from a QPSK constellation for each user. Training was performed using the Adam [30] optimizer with a batch size of 500 and a learning rate decaying from 10^{-3} to 10^{-4}.

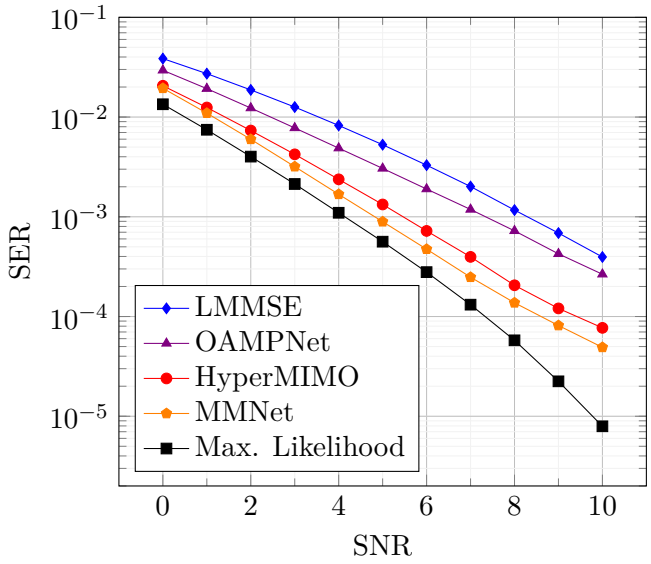


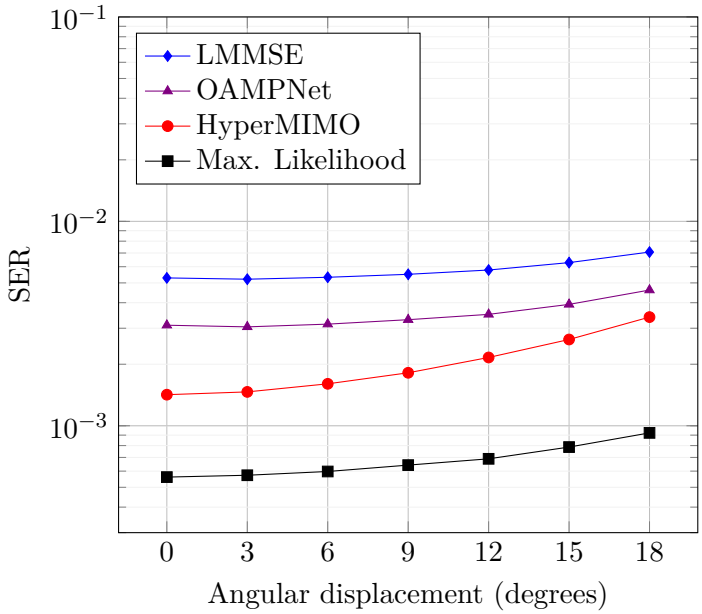
Figure 3.5: symbol error rate (SER) achieved by different schemes

3.4.3 Simulation Results

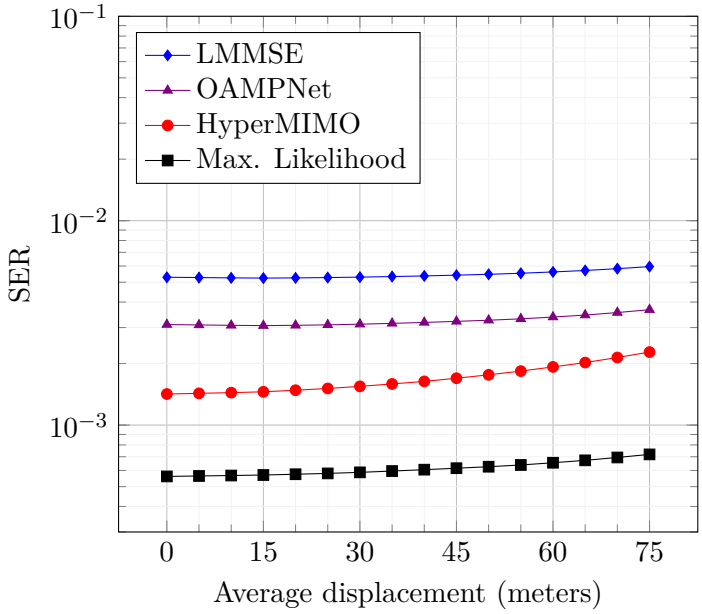
All presented results were obtained by averaging over 10 randomly generated drops of 6 users, shown in Fig. 3.4. Fig. 3.5 shows the SER achieved by HyperMIMO, LMMSE, OAMPNet with 10 iteration, MMNet with 10 iterations and trained for each channel realization, and the maximum likelihood detector. As expected, MMNet when trained for each channel realization achieves a performance close to that of maximum likelihood. One can see that the performance of OAMPNet are close to that of LMMSE on these highly correlated channels. HyperMIMO achieves SER slightly worse than MMNet, but outperforms OAMPNet and LMMSE. More precisely, to achieve a SER of 10^{-3}, HyperMIMO exhibits a loss of 0.65dB compared to MMNet, but a gain of 1.85dB over OAMPNet and 2.85dB over LMMSE.

The robustness of HyperMIMO to user mobility was tested by evaluating the achieved SER when users undergo angular mobility (Fig. 3.6a) or move in random 2D directions (Fig. 3.6b) from the positions for which the system was trained. Fig. 3.6a was generated by moving all users by a given angle, and evaluating HyperMIMO for these new users positions (and therefore new channel spatial correlation matrices) without retraining. Note that averaging was done over the two possible directions (clockwise or counterclockwise) for each user. One can see that the SER achieved by HyperMIMO gracefully degrades as the angular displacement increases, and never get worse than LMMSE nor OAMPNet.

Fig. 3.6b was generated by randomly moving the users in random 2D directions. Users were located at an initial distance of $r = 250$m. The SER was computed by averaging over 100 randomly generated displacements. As in Fig. 3.6a, the SER achieved by HyperMIMO gracefully degrades as the displacement distance increases. These results show that, despite having being trained for a particular set of user positions, HyperMIMO remains relatively robust to mobility.



(a) Angular mobility



(b) Random 2D mobility

Figure 3.6: SER achieved by the compared approaches under mobility

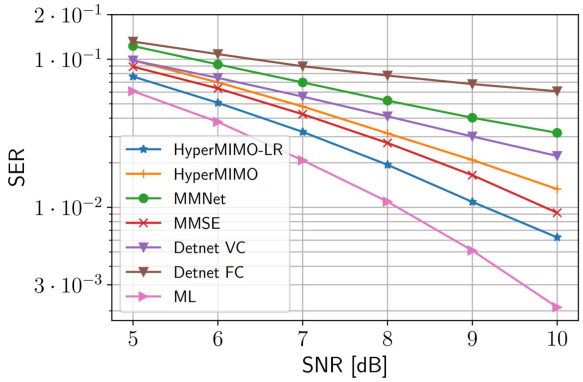


Figure 3.7: Performance comparison for different detectors, as can be found in [51]

3.5 New Perspectives and Concluding Thoughts

In order to enable performances that remain consistent across a wider set of channels, the authors in [51] proposed a variation of our HyperMIMO architecture, referred to HyperMIMO with learned regularizers, or HyperMIMO-LR. The key idea is to regularize the hypernetwork outputs with a set of MMNet parameters optimized on a single channel realization. Let us denote by $\mathbf{W}_s$ the set of MMNet parameters estimated by the hypernetwork for a given channel $\mathbf{H}_s$, i.e., $\mathbf{W}_s = \left\{\boldsymbol{\Theta}, \left[\boldsymbol{\theta}^{(1)}, \cdots, \boldsymbol{\theta}^{(I)}\right], \left[\boldsymbol{\psi}^{(1)}, \cdots, \boldsymbol{\psi}^{(I)}\right]\right\}$ when the decomposition presented in Section 3.3.1 is performed. Prior to the hypernetwork training, multiple standalone MMNet detectors are trained on a set of S channels $\mathcal{H} = \{\mathbf{H}_1, \cdots, \mathbf{H}_S\}$, resulting in a set of close-to-optimal parameters $\{\mathbf{W}_1^*, \cdots, \mathbf{W}_S^*\}$ that entail good detection performance for their corresponding channels. Then, the hypernetwork is trained to both minimize the MSE, as in (3.14), and penalize the distance between parameters estimated for the set of channel $\mathcal{H}$ and the close-to-optimal ones:

$$\mathcal{L} = \mathbb{E}_{\mathbf{x},\mathbf{H},\mathbf{n}}\left[||\widehat{\mathbf{x}} - \mathbf{x}||_2^2\right] + \beta \sum_{s=1}^{S} ||\mathbf{W}_s - \mathbf{W}_s^*||_1. \tag{3.19}$$

The parameter β allows choosing a trade-off between optimizing for a given distribution of channels and achieving high performance on a set of pre-specified channels. Moreover, the l_1 norm is leveraged in the right-hand term of (3.19) to promote sparse differences between $\mathbf{W}_s$ and $\mathbf{W}_s^*$ so that some of their entries coincide.

HyperMIMO-LR is evaluated on channels generated following a more realistic Jakes model [60] with time sequences of lengh 4, but the number of user and receive antennas were reduced to $K = 2$ and $L = 4$, respectively. The trade-off parameter was set to $\beta = 1$ and the dataset $\mathcal{H}$ contained 561 channels. Simulation results, as reported in Fig. 3.7, indicate that HyperMIMO-LG is able to provide gains compared to a standard HyperMIMO detector. One can notice that the varying channels generated from the Jakes model pose significant difficulties to all other DL-based methods, as the LMMSE detector slightly outperforms HyperMIMO,

which itself outperforms DetNet trained with varying channels (DetNet VC), MMNet, and DetNet trained with fixed channels (DetNet FC).

Although HyperMIMO-LG is reasonably close to the performance of the maximum likelihood equalizer (ML in Fig. 3.7), the fact that the detector only handles a specific number of users hinders its implementation on beyond-5G systems. To tackle this problem, a recurrent equivariant MIMO (RE-MIMO) detector has been proposed in [52]. This detector capitalizes on the recent advances in transformer networks [61] and recurrent inference networks [62] to handle a variable number of users with a single NN. An additional advantage is that the RE-MIMO detector is permutation invariant, which means that the ordering of signals received from the users has no impact on the detection performance. RE-MIMO can also be seen as an unfolded RNN, where each iterative unit consists of a module that computes the gradient of a likelihood model $P(\mathbf{y}|\mathbf{x})$ and of an encoder and predictor modules that output an updated hidden state and an estimation of the transmitted signal. Evaluations performed on correlated channels with $K = 16$ users and $L = 64$ receive antennas show tangible performance gains compared to the OAMPNet algorithm, indicating that RE-MIMO should be able to match the performance of HyperMIMO-based algorithms. However, this performance and scalability come at a complexity cost, since the attention layers used at every iteration of the unfolded RNN each contains multiple large trainable matrices.

Overall, this chapter has introduced a body of work dealing with NN-based optimization of the equalization block. More specifically, we proposed to leverage the idea of hypernetworks to alleviate the need for retraining an MMNet detector for each channel realization, while still achieving competitive performance. To reduce the complexity of the hypernetwork, MMNet was modified to decrease its number of trainable parameters, and a form of weights sharing was used. Simulations revealed that the resulting HyperMIMO architecture achieves near state-of-the-art performance under highly correlated channels when trained and evaluated with the same number of users and with fixed channel statistics. These weaknesses were subsequently addressed in more recent work, with the HyperMIMO-LR variant providing gains on a wider variety of channels, and more complex schemes such as the RE-MIMO detector being able to handle a varying number of users. However, while these NN-based detectors represent promising improvements compared to traditional algorithms, their block-based optimization still provides no guaranties on the overall receiver optimality. An additional drawback is that all the presented NN-based detectors require perfect channel estimates at training, which are usually not available in practice. For these reason, we focus in the next chapter on transceiver-based optimization, albeit for SISO systems only.

— 4 —

Learning OFDM Waveforms with PAPR and ACLR Constraints

4.1 Motivation

As discussed in Chapter 3, MU-MIMO is an efficient technique to improve the spectral efficiency when the BS is equipped with multiple antennas. However, spatial multiplexing cannot be exploited on SISO systems, and therefore other approaches must be considered. One of them is the design of new waveforms that satisfy stronger requirements on the signal characteristics. For example, a higher number of connected devices suggests that the available spectrum should be more efficiently shared among users, challenging the need for guard bands. Moreover, the power amplifier (PA) nonlinearities lead to the use of large power back-offs that decrease its efficiency, and therefore the probability of high-amplitude signals should be reduced. Among possible candidates, OFDM is already used in most modern communication systems thanks to a very efficient hardware implementation and a single-tap equalization at the receiver. However, conventional OFDM suffers from multiple drawbacks, including the need for pilots, a high sensitivity to Doppler spread, and both a high PAPR and ACLR, which might hinder its use in beyond-5G systems.

Future base stations and user equipments are expected to be equipped with dedicated DL accelerators [63], which should be eventually be able to support fully NN-based transceivers. In this chapter, we take advantage of such capabilities and review the learning-based approach to OFDM waveform design that we proposed in [64]. More specifically, this approach is based on a CNN transmitter that implements a high-dimension modulation scheme and a CNN-based

receiver that computes LLRs on the transmitted bits. Both transceivers operate on top of OFDM to benefit from its efficient hardware implementation and process OFDM symbols instead of individual REs. We derive a training procedure which allows to both maximize an achievable information rate and to offset OFDM drawbacks by defining specific optimization constraints. The end-to-end system training is performed through SGD, and therefore the achievable rate and the constraints need to be expressed as functions that can be evaluated and differentiated during training.

In the following, we focus on designing OFDM-based waveforms that enable pilotless transmissions and satisfy PAPR and ACLR constraints. The end-to-end system is benchmarked against a close to ideal implementation of a TR baseline, in which a number of subcarriers are used to generate peak-reduction signals. Both systems are evaluated on a 3rd Generation Partnership Project (3GPP)-compliant channel model, and the baseline uses a pilot configuration supported by the 5G new radio (NR) specifications. The end-to-end system, on the contrary, does not use any pilots and learns a high-dimensional modulation that enables accurate detection at the receiver. Evaluation results show that the learning-based system allows meeting PAPR and ACLR targets and enables throughput gains ranging from 3% to 30% compared to a baseline with similar characteristics. To get insight into how the proposed system reduces the PAPR and ACLR while maintaining high rates, we have carried out a detailed study of the learned high-dimensional modulation scheme. We have found out that the ACLR and PAPR reductions are achieved through a combination of spectral filtering, uneven energy allocation across subcarriers, and positional adjustments of constellation symbols. To the best of our knowledge, this method is the first DL-based approach that jointly maximizes an information rate of OFDM transmissions and allows setting PAPR and ACLR targets.

Related literature

To counteract the drawbacks of OFDM-based waveforms, previous works suggested to filter the analog signal, to modify the constellation used for modulation, or to inject custom signals on reserved subcarriers. The first approach is known as iterative clipping and filtering, and iterates between clipping in the time-domain and filtering in the frequency domain to constrain the signal amplitude and possibly the spectral leakage [65]. The second scheme, named active constellation extension, extends the outer symbols of a constellation to reduce the signal PAPR at the cost of an increased power consumption [66]. Finally, the third technique reserves a subset of available subcarriers to create a peak-cancelling signal, and is referred to as tone reservation (TR) [67].

Motivated by the success of DL in other physical layer tasks, multiple works suggested replacing existing PAPR reduction algorithms with DL components. For example, an NN was used to generate the constellation extension vectors in [68]. Similarly, a TR algorithm was unfolded as NN layers in [69]–[71]. It has also been proposed to model the communication system as an autoencoder that is trained to both minimize the PAPR and symbol error rate [72]. Although closer to our approach, the proposed scheme operates on symbols, meaning that the

bit mapping and demapping have to be implemented separately, and has only been evaluated on a simple Rayleigh fading channel. Moreover, the time-domain signal is not oversampled, which is required to obtain an accurate representation of the underlying waveform for PAPR calculations [73]. Finally, none of these works allow setting precise PAPR and ACLR targets, which means that the trade-off between the PAPR, ACLR, and spectral efficiency can not be accurately controlled.

Regarding the end-to-end training of communication systems, numerous works proposed to use this technique to design transceivers aimed at different channels. For example, the learning of constellation geometries to achieve pilotless and CP-less communication over OFDM channels was done in [16], and the design of NN-based systems for multicarrier fading channels and optical fiber communications were respectively studied in [74] and [75]. Finally, the design of transmit and receive filters for single-carrier systems under spectral and PAPR constraints was presented in [76].

4.2 Problem Statement

A SISO system using OFDM is considered ($K = L = 1$). In this section, the OFDM channel model and the expressions of the signal waveform and spectrum are recalled. The ACLR and PAPR metrics typically used to characterize the analogue signal are then detailed. Finally, a close to ideal implementation of a TR baseline is introduced, where a subset of subcarriers are reserved to minimize the signal PAPR and pilots are transmitted to estimate the channel.

4.2.1 System Model

Channel model

The OFDM channel model as derived in Section 2.1.2 is considered, with N subcarriers and one time slot, which consists of $M = 14$ OFDM symbols. In this chapter, the subcarriers are indexed by the set $\mathcal{N} = \left\{-\frac{N-1}{2}, \cdots, \frac{N-1}{2}\right\}$, with N assumed odd for convenience. When considering the entire RG, the OFDM channel of (2.22) can be expressed as

$$\mathbf{Y} = \mathbf{H} \odot \mathbf{X} + \mathbf{N} \tag{4.1}$$

where $\mathbf{X} \in \mathbb{C}^{M \times N}$ and $\mathbf{Y} \in \mathbb{C}^{M \times N}$ respectively represent the matrix of transmitted and received FBSs, $\mathbf{H} \in \mathbb{C}^{M \times N}$ is the matrix of channel coefficients, and $\mathbf{N} \in \mathbb{C}^{M \times N}$ is the additive Gaussian noise matrix such that each element has a variance σ^2. We consider a slow-varying environment so that the channel can be assumed constant over the duration of a slot. The matrix of bits to be transmitted on the OFDM symbol m is denoted $\mathbf{B}_m = [\mathbf{b}_{m,1}, \cdots, \mathbf{b}_{m,N}]^\mathsf{T}$, where $\mathbf{b}_{m,n} \in \{0,1\}^Q, m \in \{1, \cdots, M\}, n \in \mathcal{N}$, is a vector of bits to be transmitted and Q is the number of bits per channel use. The transmitter modulates each $\mathbf{B}_m$ onto the FBSs $\mathbf{x}_m \in \mathbb{C}^N$, which are mapped on the orthogonal subcarriers to form the spectrum

$$S_m(f) = \sum_{n \in \mathcal{N}} x_{m,n} \frac{1}{\sqrt{\Delta_f}} \text{sinc}\left(\frac{f}{\Delta_f} - n\right) \tag{4.2}$$

where Δ_f is the subcarrier spacing. When CPs on duration T^{CP} are prepended to the OFDM symbols of duration T, the signal spectrum becomes

$$S_m^{\text{CP}}(f) = \sum_{n \in \mathcal{N}} x_{m,n} \frac{1}{\sqrt{\Delta_f^{\text{CP}}}} \text{sinc}\left(\frac{f - n\Delta_f}{\Delta_f^{\text{CP}}}\right) \tag{4.3}$$

where $\Delta_f^{\text{CP}} = \frac{1}{T + T^{\text{CP}}}$. We recall from (2.5) that the corresponding signal is expressed as

$$s(t) = \sum_{m=0}^{M-1} s_m(t) = \sum_{m=0}^{M-1} \sum_{n_i n \mathcal{N}} x_{m,n} \phi_n(t - mT^{\text{tot}}) \tag{4.4}$$

where $T^{\text{tot}} = T + T^{\text{CP}}$ and the transmit filters $\phi_n(t), n \in \mathcal{N}$ are defined as

$$\phi_n(t) = \frac{1}{\sqrt{T^{\text{tot}}}} \text{rect}\left(\frac{t}{T^{\text{tot}}} - \frac{1}{2}\right) e^{j2\pi n \frac{t-T^{\text{CP}}}{T}}. \tag{4.5}$$

Relevant metrics

OFDM waveforms have, inter alia, two major drawbacks. The first one is their high amplitude peaks, which create distortions in the output signal due to the saturation of the PA. Such distortions are usually reduced by operating the PA with a large power back-off or by leveraging complex digital pre-distortion, thus reducing the power efficiency. Let us denote by $\nu(t) = \frac{|s(t)|^2}{\mathbb{E}[|s(t)|^2]}$ the ratio between the instantaneous and average power of a signal. We define the PAPR_ϵ as the smallest $e \geq 0$, such that the probability of $\nu(t)$ being larger than e is smaller than a threshold $\epsilon \in (0, 1)$:

$$\text{PAPR}_\epsilon \coloneqq \min e, \;\; \text{s. t. } \; P\left(\nu(t) > e\right) \leq \epsilon. \tag{4.6}$$

Setting $\epsilon = 0$ leads to the more conventional PAPR definition $\frac{\max |s(t)|^2}{\mathbb{E}[|s(t)|^2]}$. However, the maximum signal power occurs with very low probability, and therefore such a definition of the PAPR only has a limited practical interest. Relaxing ϵ to values greater than 0 allows considering more frequent, and therefore more practically relevant, power peaks.

The second drawback of OFDM is its low spectral containment. This characteristic is typically measured with the ACLR, which is the ratio between the expected out-of-band energy $\mathbb{E}_{\mathbf{x}_m}[E_{O_m}]$ and the expected in-band energy $\mathbb{E}_{\mathbf{x}_m}[E_{I_m}]$:

$$\text{ACLR} \coloneqq \frac{\mathbb{E}_{\mathbf{x}_m}[E_{O_m}]}{\mathbb{E}_{\mathbf{x}_m}[E_{I_m}]} = \frac{\mathbb{E}_{\mathbf{x}_m}[E_{A_m}]}{\mathbb{E}_{\mathbf{x}_m}[E_{I_m}]} - 1 \tag{4.7}$$

where E_{O_m}, E_{I_m}, and $E_{A_m} = E_{O_m} + E_{I_m}$ are respectively the out-of-band, in-band, and total energy of the OFDM symbol m. The in-band energy E_{I_m} is given by

$$E_{I_m} \coloneqq \int_{-\frac{N\Delta_f}{2}}^{\frac{N\Delta_f}{2}} |S_m(f)|^2 \, df = \mathbf{x}_m^H \mathbf{J} \mathbf{x}_m \tag{4.8}$$

where each element $j_{a,b}$ of the matrix $\mathbf{J} \in \mathbb{R}^{N \times N}$ is

$$j_{a,b} = \frac{1}{\Delta_f^{\text{CP}}} \int_{-\frac{N\Delta_f}{2}}^{\frac{N\Delta_f}{2}} \text{sinc}\left(\frac{f - a\Delta_f}{\Delta_f^{\text{CP}}}\right) \text{sinc}\left(\frac{f - b\Delta_f}{\Delta_f^{\text{CP}}}\right) df, \quad a, b \in \mathcal{N}. \tag{4.9}$$

The effect on the CP length on the in-band energy is shown in Fig.4.1, which have been obtained by sending 10^6 random FBSs $\mathbf{x}_m \sim \mathcal{CN}(\mathbf{0}, \frac{1}{\sqrt{2}}\mathbf{I})$ on $N = 25$ subcarriers with CP lengths of $T^{\text{CP}} \in [0, 0.1T]$. It can be seen that the in-band energy increases with the CP length, which is to be expected as increasing T^{CP} amounts to modulating the subcarriers with sinc for which the ripples are brought closer together, thus containing more energy in

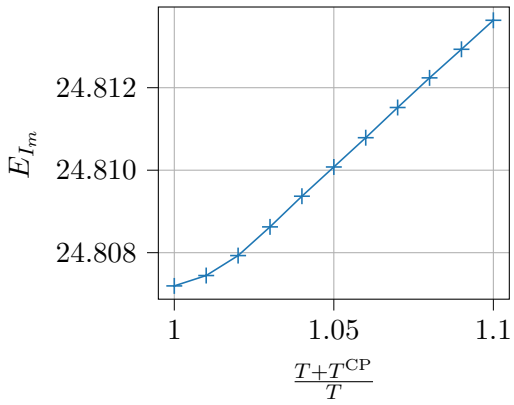


Figure 4.1: Effect of the CP length on the in-band energy.

$\left[\Delta_f\left(n-\frac{1}{2}\right), \Delta_f\left(n+\frac{1}{2}\right)\right]$. This in-band energy increase can be directly mapped to an ACLR decrease, as the total energy does not depend on CP length. In the following, we therefore consider that $T^{\mathrm{CP}} = 0$ when computing the time-domain representation and spectrum of the signal, as it corresponds to the worst-case scenario in terms of spectral energy leakage.

Finally, the total energy can be more conveniently computed in the time domain:

$$E_{A_m} \coloneqq \int_{-\frac{T}{2}}^{\frac{T}{2}} |s(t)|^2 \, dt = \mathbf{x}_m^H \mathbf{K} \mathbf{x}_m \tag{4.10}$$

where $\mathbf{K} \in \mathbb{R}^{N \times N}$ has elements

$$k_{a,b} = \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} e^{i2\pi(a-b)t/T} dt, \quad a, b \in \mathcal{N}. \tag{4.11}$$

4.2.2 Baseline

One technique to reduce the PAPR of OFDM signal is TR, in which a subset of R tones (subcarriers) is used to create peak-reduction signals. These subcarriers are referred to as peak reduction tones (PRTs), and the remaining D subcarriers are used for data and pilot transmission. The sets containing the PRTs and the data-carrying subcarriers are respectively denoted by $\mathcal{R}$ and $\mathcal{D}$, and are such that $\mathcal{R} \cup \mathcal{D} = \mathcal{N}$.

Transmitter

The TR-based transmitter sends three types of signals: data signals, peak reduction signals, and pilot signals which are used by the receiver to estimate the channel. Such pilots are inserted in the RG following the 5G NR pattern illustrated in Fig. 4.2, i.e., every two REs on the second OFDM symbol, and the value of each pilot is chosen randomly on the unit circle. For clarity, only data and peak-reduction signals are considered when describing the transmitter, as transmitting pilots is achieved by simply replacing some REs carrying data by

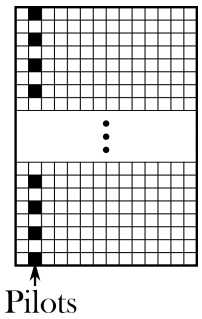


Figure 4.2: Pilot pattern used by the baseline.

reference signals. We denote by $d_{m,n\in\mathcal{D}}$ and $r_{m,n\in\mathcal{R}}$ the FBSs carrying data and peak-reduction signals, respectively. An FBS $d_{m,n\in\mathcal{D}}$ corresponds to the mapping of a vector of bits $\mathbf{b}_{m,n\in\mathcal{D}}$ following a 2^Q-QAM, the constellation of which is denoted by $\mathcal{C} \in \mathbb{C}^{2^Q}$. The vector of FBSs $\mathbf{d}_m \in \mathbb{C}^N$ is composed of all $d_{m,n\in\mathcal{D}}$ and of zeros at positions that correspond to PRTs, i.e., $d_{m,n\in\mathcal{R}} = 0$. The reduction vector $\mathbf{r}_m \in \mathbb{C}^N$ is formed by the signals $r_{m,n\in\mathcal{R}}$ mapped to the PRTs, and is conversely such that $r_{m,n\in\mathcal{D}} = 0$. As an example, if three subcarriers are used and only the last one is used as a PRT, these vectors are expressed as $\mathbf{d}_m = [d_{m,-1}, d_{m,0}, 0]^\mathsf{T}$ and $\mathbf{r}_m = [0, 0, r_{m,1}]^\mathsf{T}$. The vector of discrete baseband signals to be transmitted and the corresponding continuous-time waveform are finally denoted by $\mathbf{x}_m = \mathbf{d}_m + \mathbf{r}_m$ and $s_m(t)$, respectively. TR aims at finding $\mathbf{r}_m$ that minimizes the maximum squared signal amplitude:

$$\arg\min_{\mathbf{r}_m} \max_t |s_m(t)|^2. \tag{4.12}$$

As minimization over the time-continuous signal leads to intractable calculations, $s_m(t)$ is first discretized. Many previous studies considered a discrete vector $\mathbf{z}_m \in \mathbb{C}^N$, sampled with a period $\frac{T}{N}$, as a substitute for the underlying signal [70]–[72]. However, it has been shown that using a vector $\underline{\mathbf{z}}_m \in \mathbb{C}^{NO_S}$, oversampled by a factor O_S, is necessary to correctly represent the analog waveform [73]. The difference between the two discretized vectors are visible in Fig. 4.3, where the squared amplitude of $\mathbf{z}_m$ and $\underline{\mathbf{z}}_m$ are plotted for an arbitrary OFDM symbol, with $N = 75$ subcarriers and an oversampling factor of $O_S = 5$. It can be seen that the oversampled signal exhibits a different maximum peak with a higher amplitude as well as numerous secondary peaks that are not present in the non-oversampled signal. These peaks might lie in the PA saturation region, causing distortions of the transmitted waveform. Let us define the IDFT matrix $\mathbf{F}^\mathsf{H} \in \mathbb{C}^{NO_S\times N}$, where each element is expressed as $f_{a,b} = \frac{1}{\sqrt{N}O_S} e^{\frac{j2\pi ab}{NO_S}}$. The oversampled vector can be obtained with

$$\underline{\mathbf{z}}_m = \mathbf{F}^\mathsf{H}\mathbf{x}_m = \mathbf{F}\left(\mathbf{d}_m + \mathbf{r}_m\right). \tag{4.13}$$

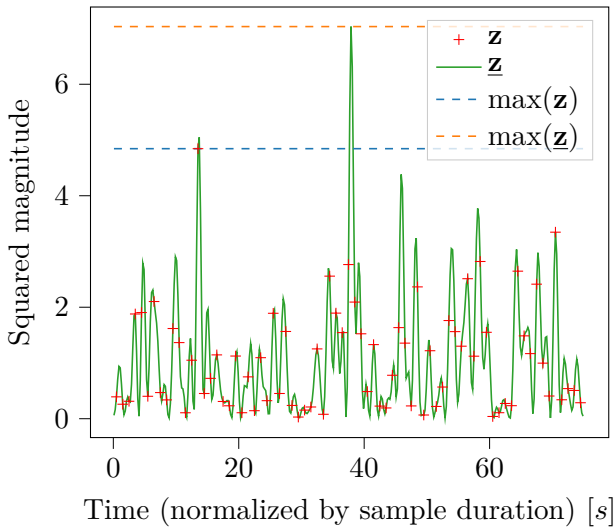


Figure 4.3: OFDM signal generated from $N = 75$ subcarriers.

The value of $\mathbf{r}_m$ that minimizes the PAPR can now be approximately found by minimizing the oversampled signal:

$$\arg\min_{\mathbf{c}_m} \left\| g\left(\mathbf{F}^{\mathsf{H}}(\mathbf{d}_m + \mathbf{r}_m)\right)\right\|_\infty \tag{4.14}$$

where $g(\cdot)$ denotes the element-wise squared magnitude $|\cdot|^2$ and $\|\cdot\|_\infty$ denotes the infinity norm. The convexity of $\|g\left(\mathbf{F}(\mathbf{d}_m + \mathbf{r}_m)\right)\|_\infty$ theoretically allows to find the optimal value of $\mathbf{r}_m$, but the associated complexity leads to the development of algorithms that approximate this value in a limited number of iterations [67]. In this thesis, however, we use a convex solver [77] to find the exact solution of (4.14) for each symbol $\mathbf{x}_m$. Although such a scheme would be prohibitively complex in practice, it is considered here to provide a close to ideal implementation of a TR-based baseline. Moreover, for fairness with conventional QAM systems, we add the convex constraint $\mathbf{r}_m^{\mathsf{H}}\mathbf{r}_m \leq R$ so that the average energy per OFDM symbol equals at most N. We experimentally verified that the average energy of the peak reduction signals $\mathbb{E}_{\mathbf{r}_m}\left[\mathbf{r}_m^{\mathsf{H}}\mathbf{r}_m\right]$ was always close to R, leading to $\mathbb{E}_{\mathbf{x}_m}\left[\mathbf{x}_m^{\mathsf{H}}\mathbf{x}_m\right] \approx N$. Finally, it was shown that placing the PRTs at random locations at every transmission leads to the lowest PAPR among other placement techniques [78]. The baseline therefore implements such a random positioning scheme for all OFDM symbols, except for the one carrying pilots for which peak-reduction signals cannot be inserted on pilot-carrying subcarriers. On this specific OFDM symbol, the number of PRTs is also reduced to $\frac{R}{2}$ in order to always maintain a significant number of subcarriers carrying data.

Receiver

On the receiver side, channel estimation is performed first, using the pilot signals received in the pilot-carrying OFDM symbol $m^{(p)} \in \mathcal{M}$. The pattern depicted in Fig. 4.2 allocates $\frac{N+1}{2}$

REs to pilot transmissions. Let us denote by $\mathbf{p}_{m^{(p)}} \in \mathbb{C}^{\frac{N+1}{2}}$ the vector of received pilot signals, extracted from $\mathbf{y}_{m^{(p)}}$. The channel covariance matrix, providing the spectral correlations between all REs carrying pilots, is denoted by $\mathbf{\Sigma} \in \mathbb{C}^{\frac{N+1}{2} \times \frac{N+1}{2}}$. This matrix can be empirically estimated by constructing a large dataset of received pilot signals and computing the statistics over the entire dataset. The channel coefficients at REs carrying pilots are estimated using an LMMSE channel estimator:

$$\widehat{\mathbf{h}}^{(p)}_{m^{(p)}} = \mathbf{\Sigma} \left(\mathbf{\Sigma} + \sigma^2 \mathbf{I}_{\frac{N+1}{2}} \right)^{-1} \mathbf{p}_{m^{(p)}} \quad \in \mathbb{C}^{\frac{N+1}{2}}. \tag{4.15}$$

Channel estimation at the remaining N REs of the OFDM symbol $m^{(p)}$ is achieved through linear interpolation. As the channel is assumed to be invariant over the duration of a slot, the so obtained vector $\widehat{\mathbf{h}}_{m^{(p)}} \in \mathbb{C}^N$ is also used for all other OFDM symbols, forming the channel estimate matrix $\widehat{\mathbf{H}} \in \mathbb{C}^{M \times N}$ where all columns are equal. On fast changing channels, pilots could be inserted in other OFDM symbols to better track the evolution of the channel.

The transmitted FBSs are estimated through equalization:

$$\widehat{\mathbf{X}} = \mathbf{Y} \oslash \widehat{\mathbf{H}}. \tag{4.16}$$

Finally, the LLR of the q^{th} bit corresponding to the RE (m, n) is computed with a conventional AWGN demapper:

$$\text{LLR}_{m,n}(q) = \ln \left(\frac{\sum_{c \in \mathcal{C}_{q,1}} \exp\left(-\frac{|\hat{h}_{m,n}|^2}{\sigma^2} |\hat{x}_{m,n} - c|^2 \right)}{\sum_{c \in \mathcal{C}_{q,0}} \exp\left(-\frac{|\hat{h}_{m,n}|^2}{\sigma^2} |\hat{x}_{m,n} - c|^2 \right)} \right) \tag{4.17}$$

where $\mathcal{C}_{q,1}$ ($\mathcal{C}_{q,0}$) is the subset of $\mathcal{C}$ containing the symbols that have the q^{th} bit set to 1 (0), and $\frac{|\hat{h}_{m,n}|^2}{\sigma^2}$ is the post-equalization noise variance.

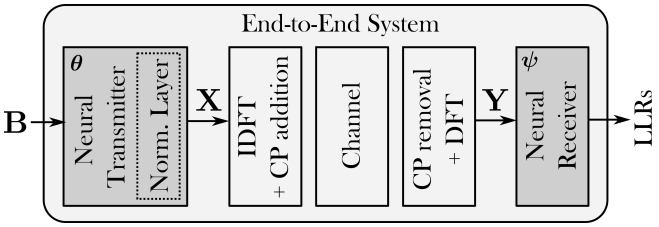


Figure 4.4: Trainable system, where grayed blocks represent trainable components.

4.3 Learning a High Dimensional Modulation

In the following, we train an NN-based transmitter and receiver to maximize an achievable rate under ACLR and PAPR constraints. This end-to-end system is referred to as "E2E" system for brevity, and is schematically shown in Fig. 4.4. An optimization procedure is derived to handle the constrained optimization problem, in which the loss function is expressed as a differentiable augmented Lagrangian. Next, we detail the transmitter and receiver architectures, both implemented as CNNs.

4.3.1 Optimization Procedure

Problem formulation

We aim at finding a high-dimensional modulation and associated detector that both maximize an information rate for the OFDM transmission and satisfy constraints on the signal PAPR and ACLR. The transmitter and receiver of the E2E system operate on top of OFDM, i.e., IDFT (DFT) is performed and a cyclic prefix is added (removed) before (after) transmission (see Fig. 4.4). The considered rate [16] is an extension of the one derived in (2.91) for an entire RG, and depends on the transmitter and receiver trainable parameters respectively denoted by $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$:

$$C(\boldsymbol{\theta},\boldsymbol{\psi}) = \frac{1}{MN}\sum_{m\in\mathcal{M}}\sum_{n\in\mathcal{N}}\sum_{q=0}^{Q-1} I\left(b_{m,n,q};\mathbf{y}_m|\,\boldsymbol{\theta}\right) \tag{4.18}$$
$$- \frac{1}{MN}\sum_{m\in\mathcal{M}}\sum_{n\in\mathcal{N}}\sum_{q=0}^{Q-1} \mathbb{E}_{\mathbf{y}_m}\left[\mathrm{D}_{\mathrm{KL}}\left(P(b_{m,n,q}|\mathbf{y}_m)||\widehat{P}_{\boldsymbol{\psi}}(b_{m,n,q}|\mathbf{y}_m)\right)\right].$$

As the E2E system outputs LLRs, the estimated posterior probabilities can be obtained from

$$\mathrm{LLR}_{m,n}(q) \coloneqq \ln\left(\frac{\widehat{P}_{\boldsymbol{\psi}}(b_{m,n,q}=1|\mathbf{y}_m)}{\widehat{P}_{\boldsymbol{\psi}}(b_{m,n,q}=0|\mathbf{y}_m)}\right). \tag{4.19}$$

To ensure a unit average energy per RE, a normalization layer is added to the transmitter

(see Fig. 4.4). Perfect normalization would perform

$$l^*_{\text{norm}}(\mathbf{x}_m) = \frac{\mathbf{x}_m}{\left(\frac{1}{N}\mathbb{E}_{\mathbf{x}_m}[E_{A_m}]\right)^{\frac{1}{2}}} \tag{4.20}$$

but the 2^{MNQ} different combinations of bits would make the computation of the expected value too complex for any practical system. Batch normalization is therefore preferred, ensuring that the average energy per RE in the batch is one:

$$l_{\text{norm}}(\mathbf{x}_m^{[j]}) = \frac{\mathbf{x}_m^{[j]}}{\left(\frac{1}{MNB_S}\sum_{m\in\mathcal{M}}\sum_{i=1}^{B_S}\mathbf{x}_m^{[i]\mathsf{H}}\mathbf{K}\mathbf{x}_m^{[i]}\right)^{\frac{1}{2}}} \tag{4.21}$$

where the superscript $[j]$ denotes the j^{th} element in the batch and B_S is the batch size. This expression is slightly different from the one typically used in related works, since it accounts for the correlation that can appear between the FBSs generated by the transmitter. Conventional bit-interleaved modulation systems produces i.i.d. symbols, and therefore does not need to take such correlation into account.

We can now formulate the constrained optimization problem we aim to solve:

$$\begin{aligned} \underset{\boldsymbol{\theta},\boldsymbol{\psi}}{\text{maximize}} \quad & C(\boldsymbol{\theta},\boldsymbol{\psi}) && \text{(4.22a)}\\ \text{subject to} \quad & \text{PAPR}_\epsilon(\boldsymbol{\theta}) = \gamma_{\text{peak}} && \text{(4.22b)}\\ & \text{ACLR}(\boldsymbol{\theta}) \leq \beta_{\text{leak}} && \text{(4.22c)} \end{aligned}$$

where γ_{peak} and β_{leak} respectively denote the target PAPR and ACLR. Note that the PAPR and ACLR depend on the transmitter parameters $\boldsymbol{\theta}$.

System training

One of the main advantages of implementing the transmitter-receiver pair as an E2E system is that it enables optimization of the trainable parameters through SGD. As seen in Chapter 2, this requires a differentiable loss function so that the gradients can be computed and backpropagated through the E2E system. In the following, the augmented Lagrangian method is leveraged to convert the problem (4.22) into its augmented Lagrangian, which acts a differentiable loss function that can be minimized with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ [79]. The key idea is to relax the constrained optimization problem into a sequence of unconstrained problems that are solved iteratively. This method is known to be more effective than the quadratic penalty method, enabling a faster and more stable convergence [80]. In the following, we express the objective (4.22a) and the constraints (4.22b) and (4.22c) as differentiable functions that can be evaluated during training and minimized with SGD.

First, the achievable rate (4.22a) can be equivalently expressed using the system binary

cross-entropy (BCE) [21], which is widely used in binary classification problems:

$$L_C(\boldsymbol{\theta}, \boldsymbol{\psi}) := -\frac{1}{MN} \sum_{m\in\mathcal{M}} \sum_{n\in\mathcal{N}} \sum_{q=0}^{Q-1} \mathbb{E}_{\mathbf{y}_m} \left[\log_2 \left(\widehat{P}_{\boldsymbol{\psi}}(b_{m,n,q}|\mathbf{y}_m) \right) \right] \tag{4.23}$$

$$= Q - C(\boldsymbol{\theta}, \boldsymbol{\psi}). \tag{4.24}$$

To overcome the complexity associated with the computation of the expected value, an approximation is typically obtained through Monte Carlo sampling:

$$L_C(\boldsymbol{\theta}, \boldsymbol{\psi}) \approx -\frac{1}{MNB_S} \sum_{m\in\mathcal{M}} \sum_{n\in\mathcal{N}} \sum_{q=0}^{Q-1} \sum_{i=0}^{B_S-1} \log_2 \left(\widehat{P}_{\boldsymbol{\psi}} \left(b_{m,n,q}^{[i]} | \mathbf{y}_m^{[i]} \right) \right). \tag{4.25}$$

Second, evaluating the constraint (4.22b) requires the computation of the probability $P(\frac{|s(t)|^2}{\mathbb{E}[|s(t)|^2]} > e)$, where e is the energy threshold defined in (4.6). However, computing such probability would be prohibitively complex due to the sheer amount of possible OFDM symbols. During training, we therefore enforce the constraint by setting $\epsilon = 0$ and penalizing all signals whose squared amplitude exceed γ_{peak}. With $\epsilon = 0$, the constraint (4.22b) is equivalent to enforcing $L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) = 0$, with

$$L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) = \mathbb{E}_m \left[\int_{-\frac{T}{2}}^{\frac{T}{2}} \left(|s_m(t)|^2 - \gamma_{\text{peak}} \right)^+ dt \right] \tag{4.26}$$

where $(x)^+$ denotes the positive part of x, i.e., $(x)^+ = \max(0, x)$. To evaluate $L_{\gamma_{\text{peak}}}(\boldsymbol{\theta})$ during training, the value of the expectation can be obtained through Monte Carlo sampling, and the integral can be approximated using a Riemann sum:

$$L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) \approx \frac{T}{B_S N O_S} \sum_{i=0}^{B_S-1} \sum_{t=-\frac{NO_S-1}{2}}^{\frac{NO_S-1}{2}} \left(\left| \underline{z}_{m,t}^{[i]} \right|^2 - \gamma_{\text{peak}} \right)^+ \tag{4.27}$$

where $\underline{\mathbf{z}}_m = \mathbf{F}^{\mathsf{H}} \mathbf{x}_m \in \mathbb{C}^{NO_S}$ is the vector of the oversampled time signal corresponding to the neural transmitter output $\mathbf{x}_m$.

Third, the inequality constraint (4.22c) can be converted to the equality constraint $\text{ACLR}(\boldsymbol{\theta}) - \beta_{\text{leak}} = -v$, where $v \in \mathbb{R}_+$ is a positive slack variable. This equality constraint is then enforced by minimizing $L_{\beta_{\text{leak}}}(\boldsymbol{\theta}) + v$, with

$$L_{\beta_{\text{leak}}}(\boldsymbol{\theta}) = \frac{\mathbb{E}[E_A]}{\mathbb{E}[E_I]} - 1 - \beta_{\text{leak}} \tag{4.28}$$

$$\approx \frac{\frac{1}{B_S} \sum_{i=0}^{B_S-1} \mathbf{x}^{[i]\mathsf{H}} \mathbf{W} \mathbf{x}^{[i]}}{\frac{1}{B_S} \sum_{i=0}^{B_S-1} \mathbf{x}^{[i]\mathsf{H}} \mathbf{V} \mathbf{x}^{[i]}} - 1 - \beta_{\text{leak}}. \tag{4.29}$$

Finally, for $\epsilon = 0$, the problem (4.22) can be reformulated as

$$\begin{aligned} \underset{\boldsymbol{\theta},\boldsymbol{\psi}}{\text{minimize}} \quad & L_C(\boldsymbol{\theta},\boldsymbol{\psi}) && \text{(4.30a)} \\ \text{subject to} \quad & L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) = 0 && \text{(4.30b)} \\ & L_{\beta_{\text{leak}}}(\boldsymbol{\theta}) + v = 0 && \text{(4.30c)} \end{aligned}$$

where the objective and the constraints are differentiable and can be estimated at training. The augmented Lagrangian method introduces two types of hyperparameters that are iteratively updated during training. The first one corresponds to the penalty parameters which are slowly increased to penalize the constraint with increasing severity. The second one corresponds to estimates of the Lagrange multipliers, as defined in [79]. Let us denote by $\mu_p > 0$ and $\mu_l > 0$ the penalty parameters and by λ_p and λ_l the Lagrange multipliers for the constraint functions $L_{\gamma_{\text{peak}}}(\boldsymbol{\theta})$ and $L_{\beta_{\text{leak}}}(\boldsymbol{\theta})$, respectively. The corresponding augmented Lagrangian is [79]

$$\begin{aligned} \overline{L}^*(\boldsymbol{\theta},\boldsymbol{\psi},\lambda_p,\lambda_l,&\mu_p,\mu_l,q) = L_C(\boldsymbol{\theta},\boldsymbol{\psi}) \\ &+ \lambda_p L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) + \frac{1}{2}\mu_p |L_{\gamma_{\text{peak}}}(\boldsymbol{\theta})|^2 \\ &+ \lambda_l \left(L_{\beta_{\text{leak}}}(\boldsymbol{\theta}) + v\right) + \frac{1}{2}\mu_l \left|L_{\beta_{\text{leak}}}(\boldsymbol{\theta}) + v\right|^2 . \end{aligned} \tag{4.31}$$

As derived in [79], the minimization of (4.31) with respect to v can be carried out explicitly for each fixed pair of $\{\boldsymbol{\theta},\boldsymbol{\psi}\}$ so that the augmented Lagrangian can be expressed as

$$\begin{aligned} \overline{L}(\boldsymbol{\theta},\boldsymbol{\psi},\lambda_p,\lambda_l,&\mu_p,\mu_l) = L_C(\boldsymbol{\theta},\boldsymbol{\psi}) \\ &+ \lambda_p L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) + \frac{1}{2}\mu_p |L_{\gamma_{\text{peak}}}(\boldsymbol{\theta})|^2 \\ &+ \frac{1}{2\mu_l}\left(\max(0, \lambda_l + \mu_l L_{\beta_{\text{leak}}}(\boldsymbol{\theta}))^2 - \lambda_l^2\right). \end{aligned} \tag{4.32}$$

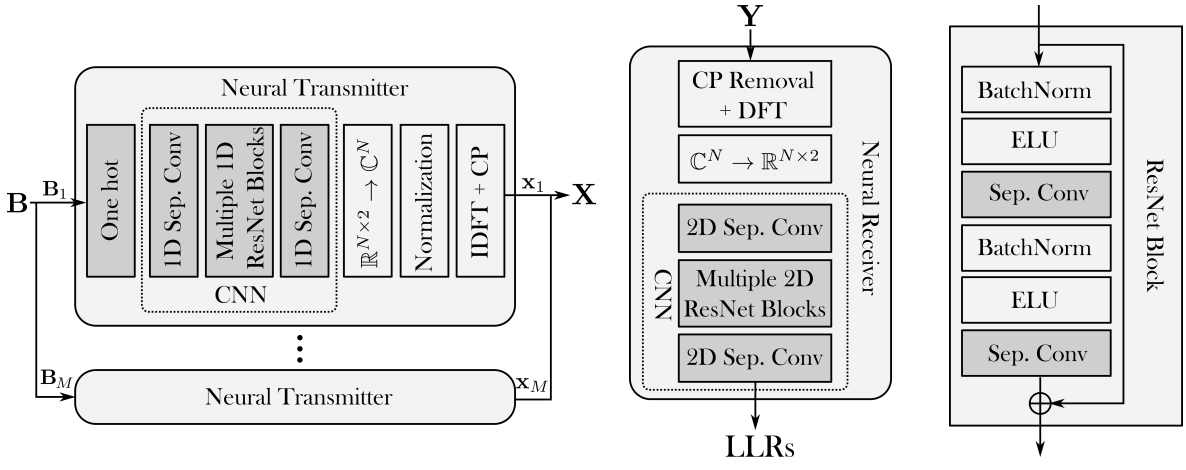
Each training iteration comprises multiples steps of SGD on the augmented Lagrangian (4.32) followed by an update of the hyperparameters. The optimization procedure is detailed in Algorithm 2, where $\tau \in \mathbb{R}^+$ controls the evolution of the penalty parameters and the superscript $u \in 0, ..., U-1$ refers to the u^{th} iteration of the algorithm.

4.3.2 System Architecture

The neural transmitter and receiver are based on similar architectures, schematically shown in Fig. 4.5. The core element is a *ResNet block*, which was introduced in the physical layer to implement a fully NN-based radio receiver [81], and whose effectiveness has been demonstrated in other related works [16], [82], [83]. A ResNet block is made of two identical sequences of layers followed by the addition of the input, as depicted in Fig. 4.5c. In the original ResNet block [84], each sequence was composed of a batch normalization layer, a rectified linear unit (ReLU) activation function, and a convolution. Our architecture differs from the original one in

Algorithm 2: Training procedure

Initialize $\boldsymbol{\theta}, \boldsymbol{\psi}, \lambda_p^{(0)}, \lambda_l^{(0)}, \mu_p^{(0)}, \mu_l^{(0)}$
for $u = 0, \cdots, U$ **do**
  ▷ Perform multiple steps of SGD
  on $\overline{L}(\boldsymbol{\theta}, \boldsymbol{\psi}, \lambda, \lambda_l, \mu_p, \mu_l)$ w.r.t. $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$
  ▷ Update optimization hyperparameters:
  $\lambda_p^{(u+1)} = \lambda_p^{(u)} + \mu_p^{(u)} L_{\gamma_{\text{peak}}}(\boldsymbol{\theta})$
  $\lambda_l^{(u+1)} = \max\left(0, \lambda_l^{(u)} + \mu_l^{(u)} L_{\beta_{\text{leak}}}(\boldsymbol{\theta})\right)$
  $\mu_p^{(u+1)} = (1+\tau)\mu_p^{(u)}$
  $\mu_l^{(u+1)} = (1+\tau)\mu_l^{(u)}$
end



(a) Neural Transmitter. (b) Neural Receiver. (c) ResNet block.

Figure 4.5: Different parts of the end-to-end system, where grayed blocks are trainable components.

that ReLUs are replaced by exponential linear units (ELUs) to alleviate the vanishing gradient problem [32] and separable convolutions are used since they enable similar performance than conventional ones but at a fraction of the computational cost [31]. Finally, we use zero-padding on all 1D (2D) convolutions to maintain constant the size of the first (and possibly second) dimension(s). In the following, the architectures of the neural transmitter and receiver are detailed, although the exact numbers of parameters for each layer are given in Section 4.4.1.

The transmitter processes all OFDM symbols in parallel (see Fig. 4.5a). Each instance of the CNN implemented at the transmitter takes as input the matrix of bits $\mathbf{B}_m$, corresponding to the OFDM symbol m, and outputs the OFDM symbol $\mathbf{x}_m$. The vector of bits is first converted into its one hot representation, i.e., into vectors of $\{0,1\}^{2^Q}$ where all elements but one are set to zero. Then, a CNN comprises one 1D separable convolution, multiple 1D ResNet blocks, and another 1D separable convolution. This CNN is fed with the one-hot matrix of dimension $N \times 2^Q$, were N corresponds to the dimension of the 1D convolution and 2^Q to different convolution channels, and outputs $N \times 2$ elements. The next layers convert these $N \times 2$ real numbers into N complex symbols and normalize them, as in (4.21), to have a unit average energy per RE. Finally, an IDFT is performed on the symbols and a CP is added before transmission. We experimentally verified that independent processing of all OFDM symbols, resulting in the use of 1D convolutions, leads to better performance than 2D convolutions that would process all OFDM symbols at once. This could be explained by the 1D nature of PAPR and ACLR measurements, which are computed for all OFDM symbols separately.

The neural receiver, on the contrary, performs a 2D processing on all OFDM symbols since it enables more accurate channel estimation and equalization [16], [83]. At reception, the CP is first removed and an DFT is applied to the received signals $\mathbf{Y}$. The $M \times N$ symbols are then converted into $M \times N \times 2$ real numbers that are fed, along with the transmission SNR of size $M \times N \times 1$, into a 2D CNN. The architecture of the receiver CNN is similar to the one of the transmitter, except that 2D separable convolutions are used. The last 2D separable layer outputs $M \times N \times Q$ real numbers that correspond to the LLRs of all transmitted symbols, as shown in Fig. 4.5b. Note that no pilots are used as it was shown in [16] that pilotless communication is possible over OFDM channels when neural receivers are used.

	Sep. Conv. 1D (2D)	ResNet blocks 1D (2D)					Sep. Conv 1D (2D)
Kernel size	1 (1,1)	3 (3,2)	9 (9,4)	15 (15,6)	9 (9,4)	3 (3,2)	1 (1,1)
Dilation rate	1 (1,1)	1 (1,1)	2 (2,1)	4 (4,1)	2 (2,1)	1 (1,1)	1 (1,1)
Filters	128						2 (Q)

Table 4.1: Parameters for the neural transmitter (receiver).

4.4 Simulations Results and Insights

In this section, the E2E system is benchmarked against the TR baseline. We first describe the training and evaluation setup, followed by rate, BER, and goodput comparisons.

4.4.1 Evaluations

Training and evaluation setup

Separate datasets were used for training and testing, both generated using a mixture of 3GPP-compliant urban microcell (UMi) line-of-sight (LOS) and non-LOS models. The channel responses were generated using QuaDRiGa 2.4.0 [85], and perfect power control was assumed such that the average channel energy per RE was one, i.e., $\mathbb{E}\left[|h_{m,n}|^2\right] = 1$. $N = 75$ subcarriers were considered with $M = 14$ OFDM symbols using CPs of sufficient lengths so that the channel can be represented by (4.1) in the frequency domain. The center carrier frequency and subcarrier spacing were respectively set to 3.5 GHz and 30 kHz, the number of bits per channel use was set to $Q = 4$, and 16-QAM modulation was used by the baseline. Coded BER comparisons were performed using a standard 5G-compliant low-density parity-check (LDPC) code with length 1024 and rate $\eta = \frac{1}{2}$. The baseline was evaluated for $R \in \{0, 2, 4, 8, 16, 32\}$ PRTs and the E2E system was trained to achieve ACLR targets of $\beta_{\text{leak}} \in \{-20, -30, -40\}$ dB and PAPR targets of $\gamma_{\text{peak}} \in \{4, 5, 6, 7, 8, 9\}$ dB.

The transmitter and receiver architectures that were used are detailed in Table 4.1. Inspired by [81], the receptive fields of the CNNs were increased using dilations, and the kernel sizes had an increasing, then decreasing number of parameters. The Lagrange multipliers were initialized to $\lambda_p^{(0)} = \lambda_l^{(0)} = 0$ and τ was set to 0.004. The penalty parameters were initialized to $\mu_p^{(0)} = 10^{-1}$ and $\mu_l^{(0)} = 10^{-3}$, which mirrors the fact that $L_{\gamma_{\text{peak}}}(\boldsymbol{\theta})$ is usually two orders of magnitude lower than $L_{\beta_{\text{leak}}}(\boldsymbol{\theta})$. The optimization procedure was composed of 2500 iterations in which 20 SGD steps were performed with a learning rate of 10^{-3} and a batch size of $B_S = 100$. The oversampling factor used to compute $\underline{\mathbf{z}}$ in (4.27) was set to $O_S = 5$, as it was

shown to be sufficient to correctly represent the analog signal [67]. Finally, the SNR

$$\mathrm{SNR} = \frac{\mathbb{E}_{h_{m,n}}\left[|h_{m,n}|^2\right]}{\sigma^2} = \frac{1}{\sigma^2} \tag{4.33}$$

was chosen randomly in the interval $[10, 30]$ dB for each RG in the batch during training. The simulations parameters are listed in Table 4.2.

Parameters	Symbol (if any)	Value
Number of OFDM symbols	M	14
Number subcarriers	N	75
Number of bits per channel use	Q	4
PAPR targets	γ_{peak}	$\{4, 5, 6, 7, 8, 9\}$ dB
ACLR targets	β_{leak}	$\{-20, -30, -40\}$ dB
PRTs used by the baseline	R	$\{0, 2, 4, 8, 16, 32\}$
Batch size	B_S	100 RGs
Oversampling factor	O_S	5
Code rate	η	$\frac{1}{2}$
Code length	-	1024 bit
Center frequency (subcarrier spacing)	-	3.5 GHz (30 kHz)
Scenario	-	3GPP 38.901 UMi LOS + NLOS
SNR	-	$[10, 30]$ dB
Learning rate	-	10^{-3}

Table 4.2: Training and evaluation parameters.

Evaluation results

In the following evaluations, the PAPR probability threshold was set to $\epsilon = 10^{-3}$. Note that setting $\epsilon = 0$ would only take into account the maximum signal peak, which is achieved by a single possible waveform that has probability $\frac{1}{2^{NQ}} \approx 10^{-90}$. The average rates per RE achieved by the baseline and the E2E systems are shown in Fig. 4.6, where the numbers next to the data points are the corresponding ACLRs. First, it can be seen that at the maximum PAPR of approximately 8.5 dB, the E2E system trained with $\beta_{\text{leak}} = -20$ dB achieves a 3% higher throughput than the baseline with no PRT. This can be explained by the rate loss due to the presence of pilots in the baseline, which do not carry data and account for approximately 4% of the total number of REs. Second, at lower PAPRs, the rates achieved by the E2E system trained with $\beta_{\text{leak}} \in \{-20, -30\}$ are significantly higher than the ones

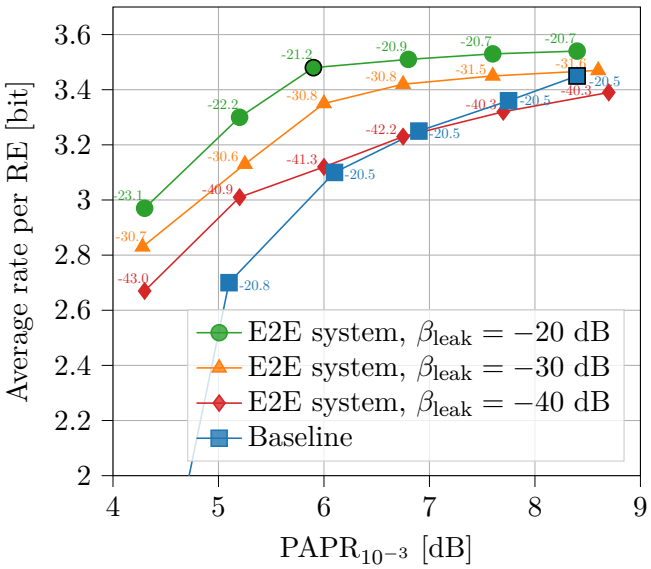


Figure 4.6: Rates achieved by the compared systems. Numbers near scatter plots indicate the ACLRs.

achieved by the baseline. For example, the E2E system trained with $\beta_{\text{leak}} = -20$ and $\gamma_{\text{peak}} = 5$ achieves an average rate 22% higher than the baseline for the same PAPR. Finally, the E2E systems are able to meet their respective PAPR and ACLR targets.

The coded BERs of the baseline and of the three E2E systems are shown in the left column of Fig. 4.7. As the baseline transmits pilots and reduction signals in addition to data signals, the energy per transmitted bit is higher than that of the of E2E systems. To reflect this characteristic, we define the energy-per-bit-to-noise-spectral-density ratio as

$$\frac{E_b}{\sigma^2} = \frac{\mathbb{E}_{x_{m,n}}\left[|x_{m,n}|^2\right]}{\rho K \sigma^2} = \frac{1}{\rho Q \sigma^2} \tag{4.34}$$

where ρ is the ratio of REs carrying data signals over the total number of REs in the RG. Fig. 4.7a, 4.7c, and 4.7e correspond to systems achieving PAPRs of approximately 8.5, 6.8, and 5.2 dB, respectively. Such PAPRs were obtained using $R = \{0, 4, 16\}$ PRT for the baseline, and $\gamma_{\text{peak}} = \{9, 7, 5\}$ dB for the trained systems. Note that evaluation results corresponding to systems trained with $\beta_{\text{leak}} = \{-30, -40\}$ dB are provided for completeness, but a fair comparison is only possible between the baseline and the system trained with $\beta_{\text{leak}} = -20$ dB as this value corresponds to the baseline ACLR. Overall, one can see that the baseline consistently achieves slightly lower BER than the E2E systems. However, the baseline also transmits fewer bits per RG, due to some REs being used to transmit pilots or reduction signals.

To understand the benefits provided by the E2E approach, the second column of Fig. 4.7 presents the *goodputs* achieved by each compared system. The goodput is defined as the

average number of information bits that have been successfully received in a RE, i.e.,

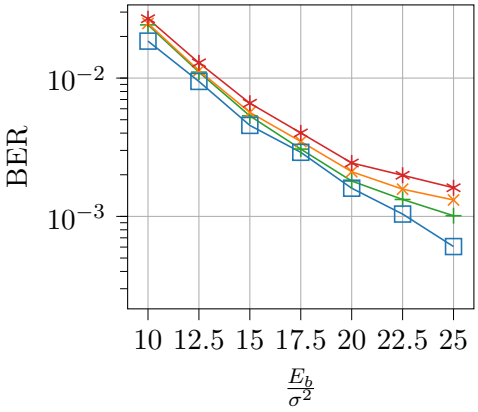
$$\text{Goodput} = \eta\rho Q(1 - \text{BER}), \tag{4.35}$$

and is plotted with respect to the SNR as it already accounts for the different number of REs transmitting information bits through the parameter ρ. Evaluation results show that the goodputs achieved by all trained systems, including those trained for lower ACLRs, are significantly higher than the ones of the baseline. Indeed, at an SNR of 30 dB, all E2E systems are able to successfully transmit close to two bits per RE, while the baseline saturates at 1.93, 1.82, and 1.53 bits for PAPRs of 8.5, 6.8, and 5.2 dB, respectively. The BER improvements range from a 3% increase at low SNR with $\text{PAPR}_\epsilon \approx 8.5$ dB to a 30% increase at high SNR with $\text{PAPR}_\epsilon \approx 5.2$ dB, indicating that the E2E approach is particularly effective when the PAPR reduction is high. These gains are jointly enabled by the pilotless nature of the E2E transmission, the effective ACLR reduction scheme learned through the proposed optimization procedure, and the fact that every REs can be used to transmit data.

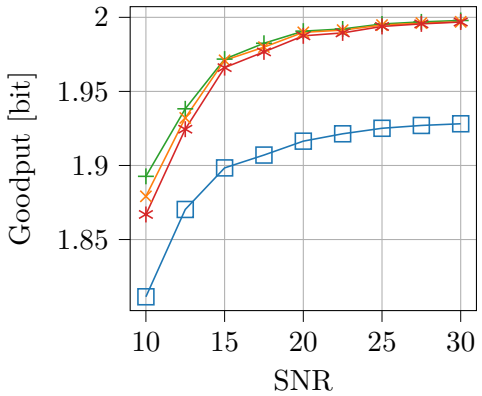
4.4.2 Insights on the Learned ACLR and PAPR Reduction Techniques

Interpreting the ACLR minimization process

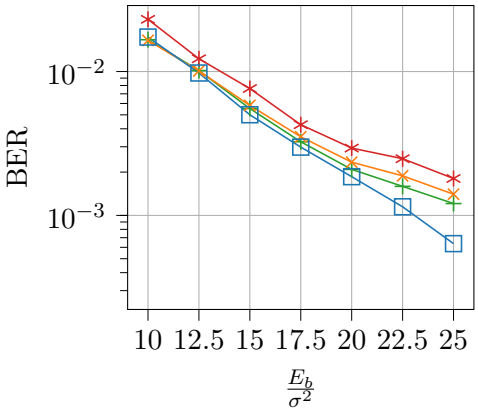
In order to get insight into the processing done by the neural transmitter, we first study the ACLR reduction technique learned by the E2E systems. Three covariance matrices $\mathbb{E}_{\mathbf{x}_m}\left[\mathbf{x}_m\mathbf{x}_m^{\mathsf{H}}\right]$ are shown in Fig. 4.8 for systems trained with $\gamma_{\text{peak}} = 9$. The first covariance matrix is from a system trained with an ACLR target of $\beta_{\text{leak}} = -20$ dB, while the second and third matrices were respectively obtained with $\beta_{\text{leak}} = -30$ dB and $\beta_{\text{leak}} = -40$ dB. It can be observed that the correlation between the elements close to the diagonal increase inversely to the ACLR target. Moreover, the correlation increases at subcarriers located near the edges of the spectrum, indicating that a subcarrier-dependent filtering is learned. Fig. 4.9 depicts the distribution of the energy across the subcarriers. On the one hand, the system trained with a lax ACLR constraint ($\beta_{\text{leak}} = -20$ dB) equally distributes the available energy on all subcarriers, which can be shown to maximize the information rate of the transmission. On the other hand, the systems trained with lower ACLR targets learn to reduce the energy of the subcarriers located at the RG edges, also contributing the out-of-band emissions reduction. The joint effect of the filtering and of the uneven energy distribution across subcarriers is visible in Fig. 4.10, where the PSDs of the compared systems are shown. One can see that the system trained with $\beta_{\text{leak}} = -20$ dB and the baseline have similar PSDs, while the PSDs of the systems trained with lower ACLR targets present significantly lower out-of-band emissions. It therefore appears that the improved ACLR allowed by the E2E system is the result of both reducing the power of the subcarriers located near the RG edges and the introduction of correlations between the transmitted symbols.



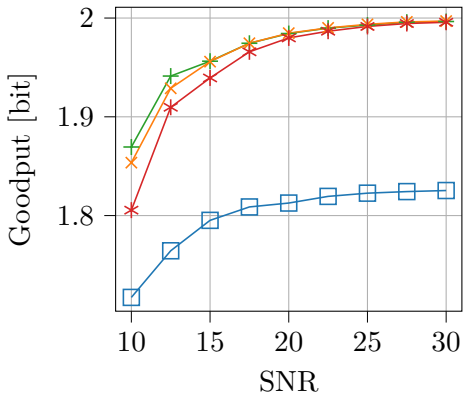
(a) Coded BER with $\gamma_{\text{peak}} = 9$ dB, $R = 0$



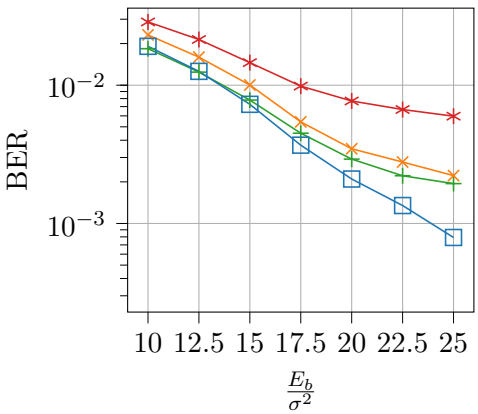
(b) Goodput with $\gamma_{\text{peak}} = 9$ dB, $R = 0$



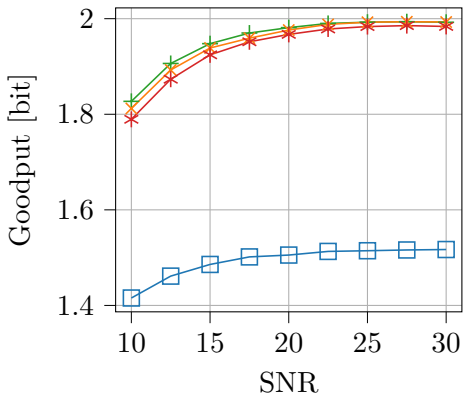
(c) Coded BER with $\gamma_{\text{peak}} = 7$ dB, $R = 4$



(d) Goodput with $\gamma_{\text{peak}} = 7$ dB, $R = 4$

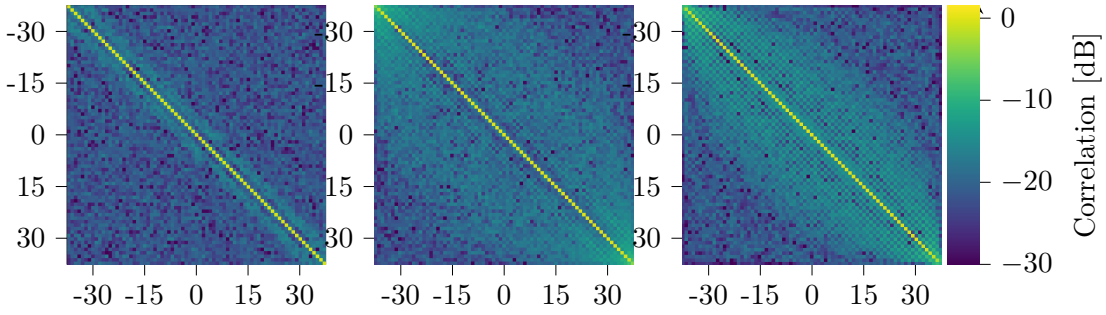


(e) Coded BER with $\gamma_{\text{peak}} = 5$ dB, $R = 16$



(f) Goodput with $\gamma_{\text{peak}} = 5$ dB, $R = 16$

Figure 4.7: BER and goodput achieved by the different schemes.



(a) $\beta_{\text{leak}} = -20$ dB (b) $\beta_{\text{leak}} = -30$ dB (c) $\beta_{\text{leak}} = -40$ dB

Figure 4.8: Covariance matrices $\mathbb{E}_m \left[\mathbf{x}_m \mathbf{x}_m^{\mathsf{H}} \right]$ for systems trained with different target ACLRs.

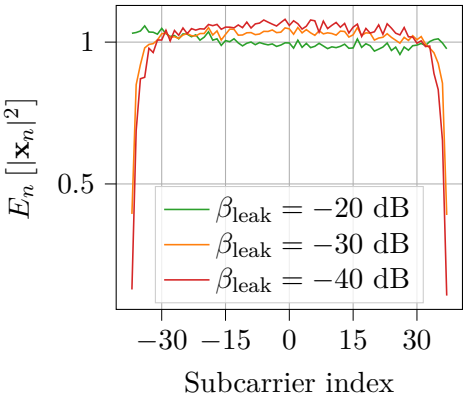


Figure 4.9: Mean energy of the symbols transmitted on each subcarrier.

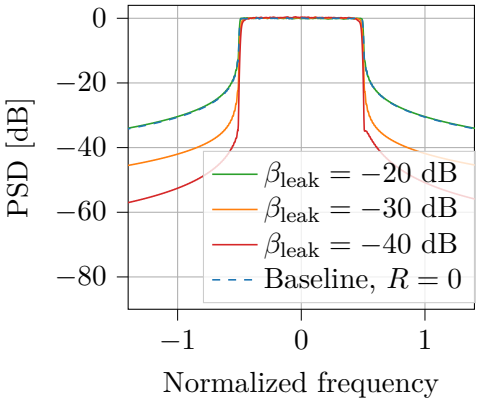


Figure 4.10: PSD of four systems having no PAPR constraint.

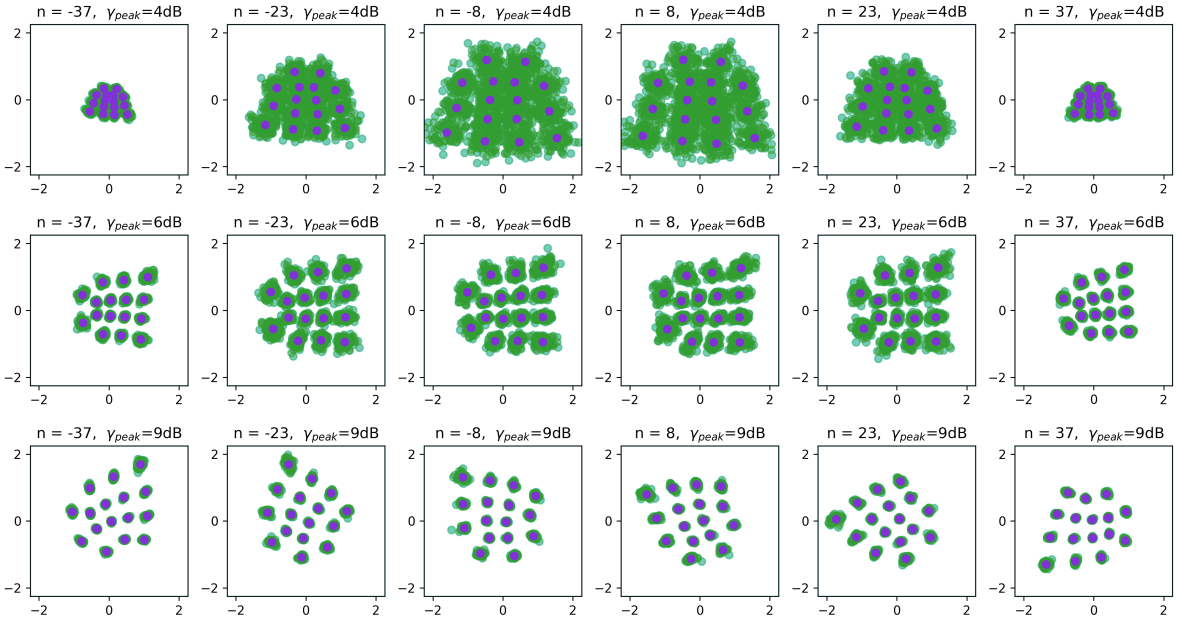


Figure 4.11: Transmitted signals on six subcarriers for E2E systems trained with different PAPR targets but a lax ACLR constraint. The purple dots represent the center of each cluster.

Interpreting the PAPR minimization process

To understand how the neural transmitter and the optimization procedure enable significant PAPR reduction, it is insightful to look at the symbols transmitted for different PAPR targets, as illustrated in Fig. 4.11. The three rows represent E2E systems trained for PAPR targets of $\gamma_{\text{peak}} \in \{4, 6, 9\}$ dB, and the six columns show the signals transmitted on the subcarriers $n \in \{-37, -23, -8, 8, 23, 37\}$. All systems where trained with a lax ACLR constraint ($\beta_{\text{leak}} = -20\text{dB}$). This figure was obtained by sending 25 RGs, and each green point represents one signal transmitted on the corresponding subcarrier. One can see that the signals are gathered into 2^Q groups, that will be referred to as *clusters* in the following. It can be observed that for low PAPR targets ($\gamma_{\text{peak}} = 4\text{dB}$), the clusters at the subcarriers located at the center of the RG exhibit more dispersion than the ones located near the RG edges. Moreover, the average energy of the transmitted signals also appears higher on the central subcarriers. On the contrary, for high PAPR targets ($\gamma_{\text{peak}} = 9\text{dB}$), the clusters seem equally dispersed and the energy is evenly spread across the subcarriers. Finally, it can be noted that the positions of the clusters are not rotationally symmetrical, which should help the neural receiver in estimating and equalizing the channel.

To better understand the neural transmitter behavior, we index each cluster by a tuple (n, k), where n is the subcarrier index and $k \in \{1, \cdots, 2^Q\}$ is the index of the cluster for the n^{th} subcarrier. Let us denote by $\left\{\mathbf{b}^{(k)}\right\}_{1 \leq k \leq 2^Q}$ the set of all possible vectors of bits indexed by their decimal representation, i.e., $\mathbf{b}^{(0)} = [0, 0]^\mathsf{T}, \mathbf{b}^{(1)} = [0, 1]^\mathsf{T}, \mathbf{b}^{(2)} = [1, 0]^\mathsf{T}$, and $\mathbf{b}^{(3)} = [1, 1]^\mathsf{T}$ for $Q = 2$. We verified that each cluster corresponds to a unique vector of bits $\mathbf{b}^{(k)}$, and the

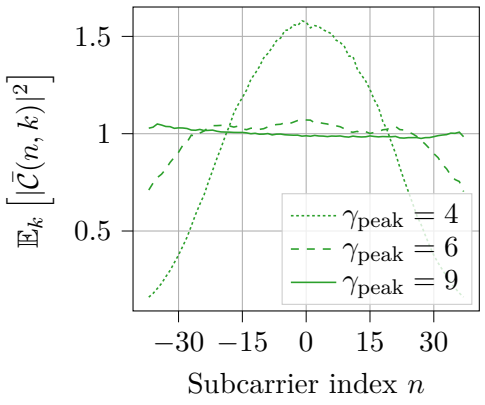


Figure 4.12: Mean energy of the clusters centers per subcarrier.

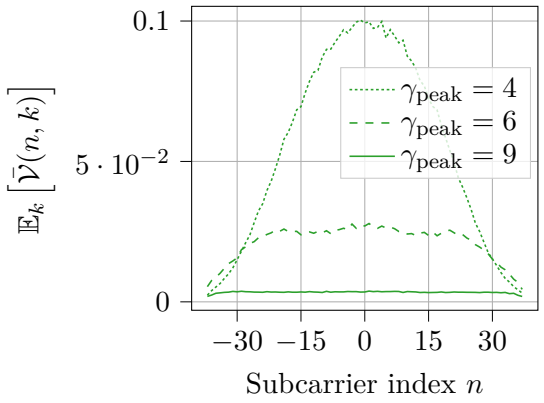


Figure 4.13: Mean variance of the clusters per subcarrier.

center of these clusters are represented by purple dots in Fig. 4.11. For each E2E system, we define a new high-dimensional constellation $\bar{\mathcal{C}} \in \mathbb{C}^{N \times 2^Q}$ that comprises the centers of the 2^Q clusters on all N subcarriers. We denote by $\bar{\mathcal{C}}(n,k) = \mathbb{E}_{\mathbf{x}_m}\left[x_{m,n}^{(k)}\right]$ the k^{th} constellation point in the n^{th} subcarrier, where $x_{m,n}^{(k)}$ denotes the output of the neural transmitter for the RE (m,n) when $\mathbf{b}^{(k)}$ was given as input. Similarly, we define $\bar{\mathcal{V}} \in \mathbb{C}^{N \times 2^Q}$, such that $\bar{\mathcal{V}}(k) = \mathbb{E}_{\mathbf{x}_m}\left[x_{m,n}^{(k)} x_{m,n}^{(k)\star}\right]$ represents the variance of the cluster (n,k).

Fig. 4.12 shows the mean energy of the constellation on each subcarrier for E2E systems trained with $\gamma_{\text{peak}} \in \{4,6,9\}$ dB and a lax ACLR constraint, corresponding to the three rows of Fig. 4.11. We can verify that when the PAPR constraint is lax ($\gamma_{\text{peak}} = 9$dB), the energy is evenly distributed across the subcarriers. On the contrary, the border subcarriers are given less energy when a harsh constraint is applied ($\gamma_{\text{peak}} = 4$dB). One explanation could be that the center subcarriers have longer wavelength, and therefore contribute less than their counterparts with shorter wavelengths. By focusing the available energy on these subcarriers, the neural transmitter jointly minimizes the probability of peaks in the analog waveform and decreases the number of FBSs that have a significant impact on them. Note that since the carrier frequency is typically much higher than the bandwidth of each subcarrier, the average power of the passband and baseband signals differs by a factor two but their maximum are approximately equal, which amounts to a roughly 3 dB difference between the passband and baseband PAPR [86]. Finally, the mean variance of the clusters on each subcarrier are plotted in Fig. 4.13 for the same three systems. One can observe that the clusters exhibit almost no variance with $\gamma_{\text{peak}} = 9$dB, but a high variance in the center frequencies with $\gamma_{\text{peak}} = 4$dB. These high variances observed with harsh PAPR constraints tend to indicate that the neural transmitter is able to slightly relocate the relevant FBSs in order to minimize the waveform PAPR. Overall, the transmitter seems to focus its energy on central subcarriers to reduce the probability of peaks and the number of relevant FBSs, and to adjust the positions of these FBSs to minimize the peak amplitudes.

In order to verify this claim, the complementary cumulative distribution function (CCDF)

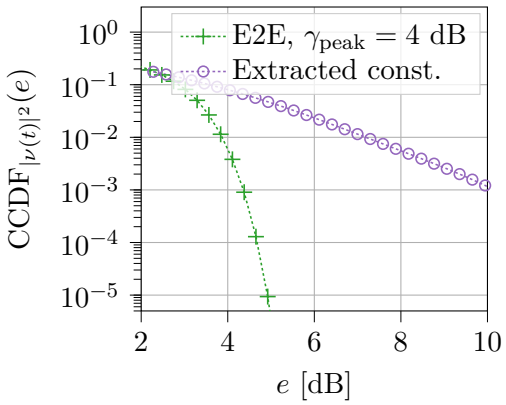


Figure 4.14: CCDF of the power of the E2E system trained for $\gamma_{\text{peak}} = 4$dB ($\beta_{\text{leak}} = -20$dB) and of a conventional system using its extracted constellation.

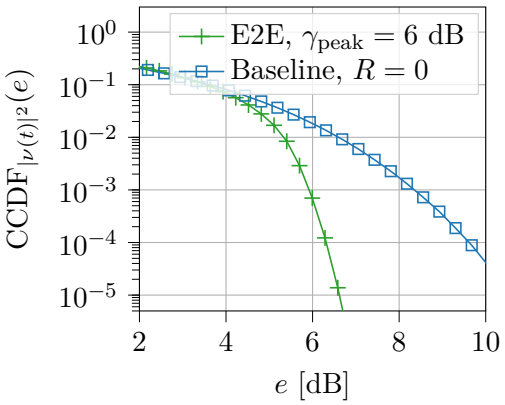


Figure 4.15: CCDF of the power of the E2E system trained with $\gamma_{\text{peak}} = 6$dB ($\beta_{\text{leak}} = -20$dB) and of the baseline using 16-QAM modulation and no PRT.

of the ratio between the instantaneous and average power of the transmitted signal is

$$\text{CCDF}_{|\nu(t)|^2}(e) = P\left(|\nu(t)|^2 > e\right) \approx P\left(\left|\frac{\underline{z}_{m,t}}{\mathbb{E}\left[\underline{z}_{m,t}\right]}\right|^2 > e\right) \tag{4.36}$$

and is approximated by sending 10000 batches of 1000 RGs, each RG i being composed of $N = 75$ subcarriers and $M = 14$ OFDM symbols $\underline{z}_{m,t}^{[i]}$ oversampled with a factor $O_S = 5$. The CCDF of two systems are presented in Fig. 4.14, the first one corresponds to the first row of Fig. 4.11, i.e., it is trained with the harshest PAPR constraint ($\gamma_{\text{peak}} = 9$dB) and a lax ACLR constraint ($\beta_{\text{leak}} = -20$dB). The second one is a conventional system that uses the constellation $\bar{\mathcal{C}}$ extracted from the aforementioned E2E system (and represented by purple dots in the first row of Fig. 4.11). The goal of this comparison is to evaluate the effect of the adjustment of the FBSs positions from the clusters centers operated by the transmitter. It can be seen that the CCDF of the E2E system is drastically lower than the one of the conventional system using the extracted constellation, indicating that the PAPR reduction is indeed performed through the FBSs position adjustments.

Finally, the CCDFs of the E2E system trained with $\gamma_{\text{peak}} = 6$dB (and a lax ACLR constraint) and of a conventional QAM system are compared in Fig. 4.15. The E2E system corresponds to the second row of Fig. 4.11, and the two compared systems are respectively highlighted by a black circle and a black square in Fig. 4.6. On can see that the PAPR minimization process operated by the neural transmitter is particularly effective, as the CCDF of the E2E system is significantly lower despite the two systems enabling similar rates. Finally, it is also interesting to note that the CCDF of the system using the extracted constellation in Fig. 4.14 is higher than the one of the 16-QAM system in Fig. 4.15. This indicates that the underlying learned constellation alone has worse PAPR characteristics than a standard QAM, demonstrating the efficiency of the positional adjustments enabled by the neural transmitter.

4.5 Concluding Thoughts

In this chapter, we proposed a learning approach to design OFDM waveforms that meet specific constrains on the envelope and spectral characteristics. We leveraged the end-to-end strategy to model the transmitter and receiver as two CNNs that perform high-dimensional modulation and demodulation, respectively. The associated training procedure first requires all optimization constraints to be expressed as differentiable functions that can be minimized through SGD. Then, a constrained optimization problem is formulated and solved using the augmented Lagrangian method. We evaluated the proposed approach on the learning of OFDM waveforms that maximize an information rate of the transmission while satisfying PAPR and ACLR constraints. Simulations were performed using 3GPP-compliant channel models, and results show that the optimization procedure is able to design waveforms that satisfy the PAPR and ACLR constraints. Moreover, the end-to-end system enables up to 30% higher throughput than a close to optimal implementation of a TR baseline with similar ACLR and PAPR.

Evaluation insights revealed that the neural transmitter achieves PAPR and ACLR reduction through a subcarrier-dependent filtering, an uneven energy distribution across subcarrier, and a positional readjustment of each constellation point. On the other side, the neural receiver is able to equalize the channel without any pilot transmitted thanks to the learned asymmetrical constellations. This directly translates into throughput gains as the associated overhead is removed. It is interesting to note that all these improvements are possible because the communication system is entirely *designed by DL*. Current standards however require the use of well-known modulations and pilot patterns to ensure compatibility across a wide range of hardware. Moreover, commercially available devices are not yet optimized to perform NN inferences at speeds that coincide with the processing time of physical layer tasks. Short-term implementations of end-to-end systems are therefore unlikely, and more practical solutions must be derived to unlock the potential of DL in the near future.

— 5 —

DL-enhanced Receive Processing for MU-MIMO Systems

5.1 Motivations

Both the block-based and end-to-end optimization strategies present advantages and drawbacks that were respectively discussed in Chapter 3 and 4. On the one hand, the block-based strategy for MU-MIMO detection has the advantages of remaining interpretable and scalable to any number of users, but is not trained to optimize the end-to-end performance and requires perfect CSI that is not available in practice. On the other hand, the end-to-end strategy allows the emergence of new waveforms that can be tailored for specific needs, but NN-based transceivers are computationally expensive and acts as black-boxes that are not suited for MU-MIMO transmissions. The quest for practical, standard-compliant, and reasonably complex designs for DL-based MU-MIMO systems therefore remains an active research topic.

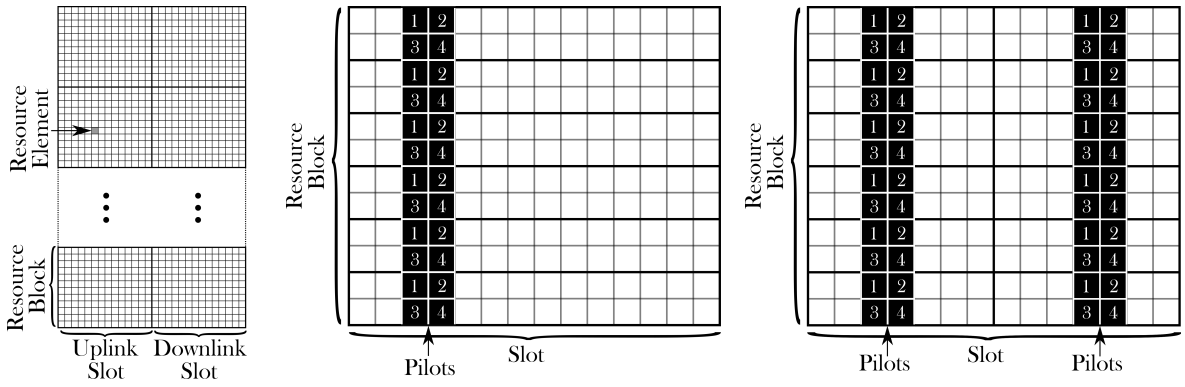
In this context, we introduced a new hybrid approach in [83], in which multiple DL components are trained in an end-to-end manner to enhance a conventional MU-MIMO receiver. The goal is to combine the interpretability of the first strategy, the efficiency of the second one, and the flexibility of traditional receivers. The resulting architecture is easily scalable to any number of users and is composed of components that are individually interpretable and of reasonable complexity. Moreover, the end-to-end training alleviates the need for perfect channel estimates. Our solution exploits the OFDM structure to counteract two known drawbacks of conventional receivers that are overlooked in the current literature. First, it improves the prediction of the channel estimation error statistics, that can only be

obtained for the pilot signals in a standard receiver, resulting in largely sub-optimal detection accuracy. Motivated by the recent successes of CNNs in physical layer tasks [54], [81], we use CNNs to predict such error statistics for every RE in the OFDM grid. In contrast to the traditional approach that is based on mathematical models, the proposed CNNs learn these statistics from the data during training. The second improvement is a CNN-based demapper that operates on the entire OFDM grid, unlike traditional demappers that operate on individual REs. In doing so, our demapper is able to better cope with the residual distortions of the equalized signal. The resulting DL-enhanced receiver is optimized such that all DL components are jointly trained to maximize the information rate of the transmission [21].

The proposed architecture is evaluated on 3GPP-compliant channel models with two different pilot configurations supported by the 5G NR specifications. Both uplink and downlink transmissions are studied using time-division duplexing (TDD). We compare the coded BER achieved by different schemes for user speeds ranging from 0 to 130 $\mathrm{km\,h^{-1}}$. Two baselines were implemented, the first one being a traditional receiver implementing a LMMSE channel estimation and an LMMSE equalizer. The second baseline also uses LMMSE equalization, but is provided with perfect channel knowledge at pilot positions and the exact second order statistics of the channel estimation errors. Our results show that the gains provided by the DL-enhanced receiver increase with the user speed, with small BER improvements at low speeds and significant ones at high speeds. We have also observed that the gains in the uplink are more pronounced than in the downlink, due to channel aging which significantly penalizes the downlink precoding scheme.

Related literature

Multiple recent papers proposed to use one large NN to jointly perform multiple processing steps. This idea has first been proposed in [87] to perform joint channel estimation and equalization in a SISO setup. This work has then been extended by additionally learning the demapper and operating directly on time-domain samples [82], [88]. The DeepRX architecture presented in [81] shows impressive results on single-input multiple-output (SIMO) channels while being 5G compliant. Another standard-compliant receiver has been proposed for Wi-Fi communications using both synthetic and real-world data [89]. Regarding MIMO transmissions, a special form of RNN called reservoir computing has been leveraged in [90] to process time-domain OFDM signals. The DeepRX receiver has also been extended with a so-called transformation layer to handle MIMO transmissions [82]. Their CNN-based solution is fed with frequency-domain signals and outputs LLRs for the transmitted bits. DeepRX MIMO shows important gains on single-user (SU)-MIMO channels, but remains very computationally expensive. Compared to our solution, the main disadvantages of these NN-based MIMO receivers are their lack of scalability and interpretability. They are tailored for a specific number of users or transmit antennas, and the CSI needed for downlink precoding can not be easily extracted.



(a) Resource grid. (b) 1P pilot pattern. (c) 2P pilot pattern.

Figure 5.1: Pilots are arranged on the RG according to two different patterns, where each number corresponds to a different transmitter.

5.2 System Model

We consider a MU-MIMO system, as presented in Section 2.1.4, in which K single-antenna users communicate with a BS equipped with L antennas in the uplink and downlink. OFDM transmissions are considered, and the RG is divided into resource blocks consisting of twelve adjacent subcarriers (Fig. 5.1a). 2^Q-QAM modulations are used to transmit data. This section introduces the channel model and the baselines against which the proposed approach is benchmarked.

5.2.1 Channel Model

Following the notations of Section 2.1.4, the channel coefficients form a 4-dimensional tensor denoted by $\mathbf{H} \in \mathbb{C}^{2M \times N \times L \times K}$, such that $\mathbf{H}_{m,n} \in \mathbb{C}^{L \times K}$ is the channel matrix at RE (m, n), and $\mathbf{h}_{m,n,k} \in \mathbb{C}^L$ is the channel vector at RE (m, n) and for user k. Duplexing is achieved through TDD, such that a slot is either assigned to the uplink or downlink in an alternating fashion, as illustrated in Fig. 5.1a. More precisely, the first slot is assigned to the uplink and the second slot is assigned to the downlink. It is assumed that channel reciprocity holds, i.e., $\mathbf{H}_{m,n}$ refers as well to the uplink or the downlink channel. To enable channel estimation, a transmitter sends pilot signals on dedicated REs according to a predefined pilot pattern. We assume, without loss of generality, that all pilots are equal to one. Two different pilot patterns are considered, referred to as the 1P and 2P pilot patterns, which respectively contain pilots on two or four symbols within a slot. Fig. 5.1b and 5.1c respectively show the 1P and 2P pilot patterns over a resource block assuming 4 users. The set of REs carrying pilots for a user $k \in \{1, \dots, K\}$ is denoted by $\mathcal{P}^{(k)}$ and the numbers of symbols and subcarrier carrying pilots are respectively denoted by $|\mathcal{P}_M|$ and $|\mathcal{P}_N|$. As an example, if the 1P pattern shown in Fig. 5.1b is used with $N = 12$, the positions $(\text{symbol}, \text{subcarrier})$ of all REs carrying pilots for user 1 are denoted by $\mathcal{P}^{(1)} = \{(3, 1), (3, 3), (3, 5), (3, 73), (3, 93), (3, 11)\}$, resulting in $|\mathcal{P}_M| = 1$ and $|\mathcal{P}_N| = 6$. Note that when a RE is allocated to a user for the transmission of a pilot,

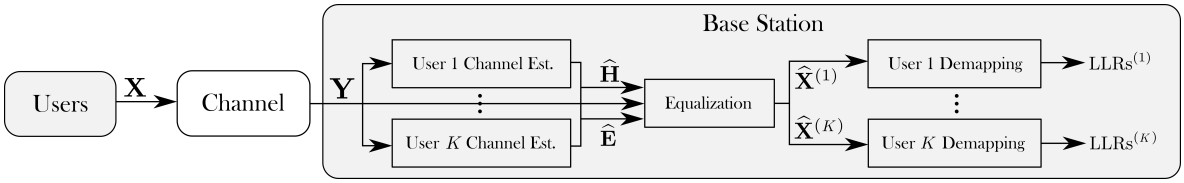


Figure 5.2: Architecture of the uplink communication system.

other users do not transmit any signal (data nor pilot) on that RE. As a consequence, pilots do not experience any interference. The noise power is denoted by σ^2 and assumed equal for all users and all REs. In the following, perfect power control is assumed over the RG such that the mean energy corresponding to a single BS antenna and a single user is one, i.e., $\mathbb{E}\left[|h_{m,n,l,k}|^2\right] = 1$. The SNR of the transmission is defined as

$$\text{SNR} = 10\log\left(\frac{\mathbb{E}\left[|h_{m,n,l,k}|^2\right]}{\sigma^2}\right) = 10\log\left(\frac{1}{\sigma^2}\right) \text{ [dB]}. \tag{5.1}$$

5.2.2 Uplink Baseline

In uplink, the BS aims to recover the bits transmitted simultaneously by the K users on the REs carrying data. The tensors of transmitted and received signals of all users are respectively denoted by $\mathbf{X} \in \mathbb{C}^{2M\times N\times K}$ and $\mathbf{Y} \in \mathbb{C}^{2M\times N\times L}$, and the transfer function on the uplink is $\mathbf{y}_{m,n} = \mathbf{H}_{m,n}\mathbf{x}_{m,n} + \mathbf{n}_{m,n}$, where $\mathbf{n}_{m,n} \sim \mathcal{CN}\left(\mathbf{0}, \sigma^2\mathbf{I}_L\right)$ is the noise vector. In this scenario, only the uplink slot is used and therefore all signals with indices $m > M$ are ignored, i.e., the corresponding values are set to 0. The architecture of the uplink system is shown in Fig. 5.2, where the IDFT (DFT) operation and the addition (removal) of the cyclic prefix before (after) the channel are not shown for clarity. The channel estimation, equalization, and demapping stages of the baseline will be explained in the following.

Channel estimation

As the pilots are assumed to be orthogonal, LMMSE channel estimation can be carried out for each user independently. The channel covariance matrix providing the spatial, temporal, and spectral correlations between all REs carrying pilots is denoted by $\mathbf{\Sigma} \in \mathbb{C}^{|\mathcal{P}_M|\cdot|\mathcal{P}_N|\cdot L\times|\mathcal{P}_m|\cdot|\mathcal{P}_n|\cdot L}$. In the following, it is assumed that the precise local statistics of the receivers are not available. The channel and receiver statistics are therefore averaged over the entire cell, resulting in a zero-mean channel, and a discussion on how these statistics can be obtained is provided in Section 5.2.4. For a user $k \in \{1,\ldots,K\}$, the LMMSE channel estimate at REs carrying pilots is denoted by $\widehat{\mathbf{H}}^{(k)}_{\mathcal{P}^{(k)}} \in \mathbb{C}^{|\mathcal{P}_M|\times|\mathcal{P}_N|\times L}$ and given by

$$\text{vec}\left(\widehat{\mathbf{H}}^{(k)}_{\mathcal{P}^{(k)}}\right) = \mathbf{\Sigma}\left(\mathbf{\Sigma} + \sigma^2\mathbf{I}_{|\mathcal{P}_M||\mathcal{P}_N|L}\right)^{-1}\text{vec}\left(\mathbf{Y}^{(k)}_{\mathcal{P}^{(k)}}\right) \tag{5.2}$$

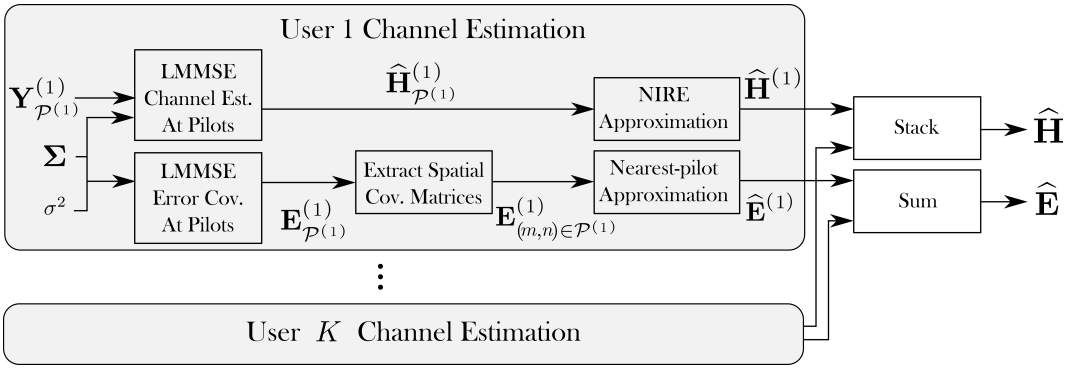


Figure 5.3: Uplink channel estimation.

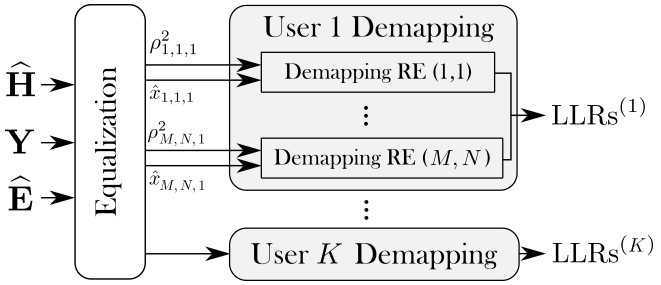


Figure 5.4: Uplink demapping

where $\mathbf{Y}^{(k)}_{\mathcal{P}^{(k)}} \in \mathbb{C}^{|\mathcal{P}_M|\times|\mathcal{P}_N|\times L}$ is the tensor of received pilots for user k. Channel estimation could also be performed at REs carrying data [16], but this would require knowledge of the channel statistics at those REs, which are typically not available in practice.

Inspired by the 3GPP guidelines [91], the channel estimates for all REs are computed by first linearly interpolating the estimates from REs carrying pilots in the frequency dimension and then using the estimate at the nearest interpolated resource element (NIRE) on the neighboring REs. It is also possible to leverage temporal linear interpolation between the OFDM symbols carrying pilots when the 2P pilot pattern is used. The so-obtained tensor of channel estimates is denoted by $\widehat{\mathbf{H}}^{(k)} \in \mathbb{C}^{2M\times N\times L}$. The overall channel estimation for all users $\widehat{\mathbf{H}} \in \mathbb{C}^{2M\times N\times L\times K}$ is obtained by stacking the channel estimates of all users. Since only the uplink slot is considered here, the channel estimates for the last M symbols (downlink slot) are set to be null. The channel estimation error is denoted by $\widetilde{\mathbf{H}}$ and is such that $\mathbf{H} = \widehat{\mathbf{H}} + \widetilde{\mathbf{H}}$. For a RE (m, n), we define

$$\mathbf{E}_{m,n} \coloneqq \mathbb{E}\left[\widetilde{\mathbf{H}}_{m,n}\widetilde{\mathbf{H}}^{\mathsf{H}}_{m,n}\right] \tag{5.3}$$

as the sum of the *spatial* channel estimation error covariance matrices from all users.

Equalization

The LMMSE equalizer derived in Chapter 3 is used, except that perfect channel estimation is not assumed anymore. Moreover, as computing a dedicated LMMSE operator for each RE is

infeasible in practice due to prohibitive complexity, we resort to a grouped-LMMSE equalizer, i.e., a single LMMSE operator is applied to a group of adjacent REs spanning multiple symbols $m \in \{M_b, \ldots, M_e\}$ and subcarriers $n \in \{N_b, \ldots, N_e\}$. Under this assumption, the LMMSE operator for the group of REs spanning $\{M_b, \ldots, M_e\} \times \{N_b, \ldots, N_e\}$ is

$$\mathbf{W}_{m,n} = \left(\sum_{m'=M_b}^{M_e} \sum_{n'=N_b}^{N_e} \widehat{\mathbf{H}}_{m',n'}^{\mathsf{H}} \right) \left(\sum_{m'=M_b}^{M_e} \sum_{n'=N_b}^{N_e} \widehat{\mathbf{H}}_{m',n'} \widehat{\mathbf{H}}_{m',n'}^{\mathsf{H}} + \mathbf{E}_{m',n'} + \sigma^2 \mathbf{I}_L \right)^{-1} \tag{5.4}$$

where $\mathbf{W}_{m,n} \in \mathbb{C}^{K \times L}$ is constant over $\{M_b, \ldots, M_e\} \times \{N_b, \ldots, N_e\}$ (see Appendix A).

The post-equalization channel is expected to be an additive noise channel. More precisely, for any RE (m, n), the demapper expects the output of the equalizer $\hat{\mathbf{x}}_{m,n} \in \mathbb{C}^K$ to be such that $\hat{\mathbf{x}}_{m,n} = \mathbf{x}_{m,n} + \mathbf{n}'_{m,n}$ where $\mathbf{n}'_{m,n}$ is an additive noise term. However, this decomposition is not achieved by the LMMSE equalizer (see Section 1.6.1 of [92] for a more detailed discussion). To obtain such a post-equalized channel, the following diagonal matrix is applied to the output of the LMMSE equalizer

$$\mathbf{D}_{m,n} = \left(\left(\mathbf{W}_{m,n} \hat{\mathbf{H}}_{m,n} \right) \odot \mathbf{I}_K \right)^{-1} \tag{5.5}$$

which re-scales the equalizer output so that the post-equalization SNR remains maximized. For a RE $(m, n) \in \{M_b, \ldots, M_e\} \times \{N_b, \ldots, N_e\}$, the equalized vector $\hat{\mathbf{x}}_{m,n}$ is computed by

$$\hat{\mathbf{x}}_{m,n} = \mathbf{D}_{m,n} \mathbf{W}_{m,n} \mathbf{y}_{m,n}. \tag{5.6}$$

The equalized symbols of user k are denoted by $\widehat{\mathbf{X}}^{(k)} \in \mathbb{C}^{M \times N}$, as shown in Fig. 5.2.

Demapping

For a RE (m, n), let us denote by $\mathbf{w}_{m,n,k}$ the column vector made of the k^{th} line of the matrix $\mathbf{W}_{m,n}$ and by $\mathbf{H}_{m,n,-k}$ the tensor made of the channel coefficients of all users except user k. After equalization, the uplink channel can be viewed as MNK parallel additive noise channels that can be demodulated independently for every RE and every user. For a RE (m, n) and user k, the post-equalization channel is expressed as

$$\hat{x}_{m,n,k} = x_{m,n,k} + \underbrace{\frac{\mathbf{w}_{m,n,k}^{\mathsf{T}} \left(\widehat{\mathbf{H}}_{m,n,-k} \mathbf{x}_{m,n,-k} + \widetilde{\mathbf{H}}_{m,n} \mathbf{x}_{m,n} + \mathbf{n}_{m,n} \right)}{\mathbf{w}_{m,n,k}^{\mathsf{T}} \widehat{\mathbf{h}}_{m,n,k}}}_{\zeta_{m,n,k}} \tag{5.7}$$

where the noise $\zeta_{m,n,k}$ includes both the interference and the noise experienced by user k. Its variance is given by

$$\begin{aligned} \rho_{m,n,k}^2 &= \mathbb{E}\left[\zeta_{m,n,k}^* \zeta_{m,n,k} \right] \\ &= \frac{\mathbf{w}_{m,n,k}^{\mathsf{H}} \left(\widehat{\mathbf{H}}_{m,n,-k} \widehat{\mathbf{H}}_{m,n,-k}^{\mathsf{H}} + \mathbf{E}_{m,n} + \sigma^2 \mathbf{I}_L \right) \mathbf{w}_{m,n,k}}{\mathbf{w}_{m,n,k}^{\mathsf{H}} \widehat{\mathbf{h}}_{m,n,k} \widehat{\mathbf{h}}_{m,n,k}^{\mathsf{H}} \mathbf{w}_{m,n,k}}. \end{aligned} \tag{5.8}$$

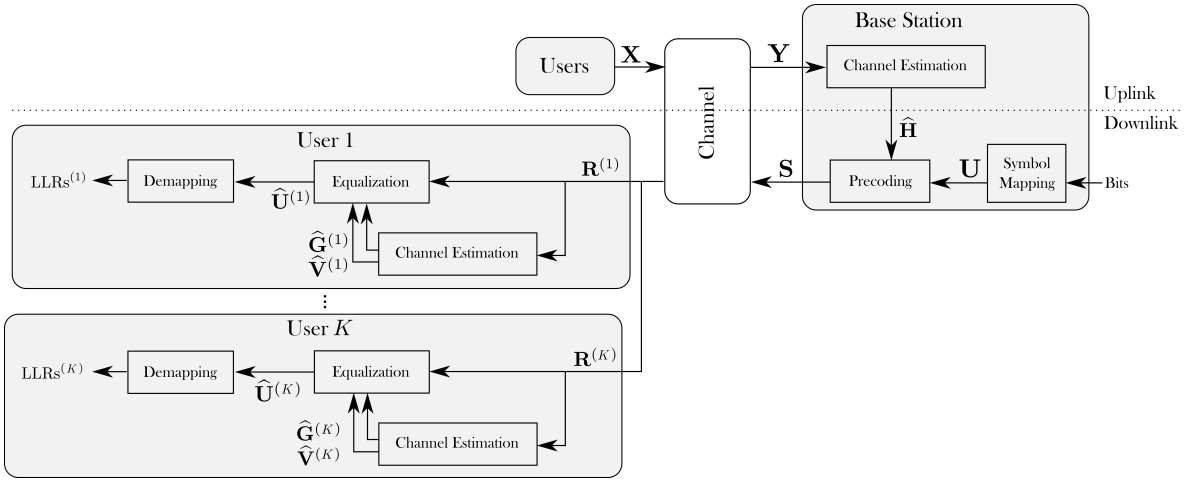


Figure 5.5: Architecture of the downlink communication system.

We denote by $\mathcal{C}_{q,0}$ ($\mathcal{C}_{b,1}$) the subset of $\mathcal{C}$ which contains all symbols with the q^{th} bit set to 0 (1). Assuming that the noise $\zeta_{m,n,k}$ is Gaussian[1], the LLRs of the q^{th} bit transmitted by user k on the RE (m,n) is given by

$$\text{LLR}^{\text{UL}}_{m,n,k}(q) = \ln\left(\frac{\sum_{c\in\mathcal{C}_{q,1}} \exp\left(-\frac{1}{\rho^2_{m,n,k}}\left|\hat{x}_{m,n,k}-c\right|^2\right)}{\sum_{c\in\mathcal{C}_{q,0}} \exp\left(-\frac{1}{\rho^2_{m,n,k}}\left|\hat{x}_{m,n,k}-c\right|^2\right)}\right). \tag{5.9}$$

The equalization and demapping process is schematically shown in Fig. 5.4.

5.2.3 Downlink Baseline

The BS aims to simultaneously transmit to K users on all REs of the downlink slot. The signal transmitted by the BS is precoded to mitigate interference. We remind that downlink transmissions occur after the uplink slot, as shown in Fig. 5.1a. Let us denote by $\mathbf{S} \in \mathcal{C}^{2M\times N\times K}$ and by $\mathbf{T} \in \mathbb{C}^{2M\times N\times L}$ the tensors of unprecoded and precoded symbols, respectively. We denote by $\mathbf{U} \in \mathbb{C}^{2M\times N\times K}$ the tensor of symbols received by the K users. Those quantities are only relevant on the downlink slot and therefore are considered null on the first M symbols, i.e., $\mathbf{s}_{m,n} = \mathbf{t}_{m,n} = \mathbf{u}_{m,n} = \mathbf{0} \quad \forall (m,n) \in \{1,\dots,M\} \times \{1,\dots,N\}$. The downlink transfer function of the channel for a RE (m,n) is

$$\mathbf{u}_{m,n} = \mathbf{H}^{\mathsf{H}}_{m,n}\mathbf{t}_{m,n} + \mathbf{q}_{m,n} \tag{5.10}$$

where $\mathbf{q}_{m,n} \sim \mathcal{CN}(\mathbf{0}, \sigma^2\mathbf{I}_K)$ is the noise vector, considered null in the first M symbols. For convenience, the noise variance σ^2 is assumed to be the same as in the uplink. Fig. 5.5 shows the architecture of the downlink system, where the IFFT (FFT) operation and the addition (removal) of the cyclic prefix before (after) the channel are again not shown for clarity. In the

[1]This is not true in general as the interference and channel estimation errors are not Gaussian distributed.

rest of this section, we detail the downlink precoding, channel estimation and equalization, and demapping steps.

Precoding

Precoding requires estimation of the downlink channel. As TDD is used, we can exploit channel reciprocity so that the downlink channel can be estimated using the nearest-pilot approach, i.e., $\widehat{\mathbf{H}}_{M+1\leq m\leq 2M,n} = \widehat{\mathbf{H}}_{M,n}$. Precoding is achieved by exploiting the uplink-downlink duality [93], which results in using $\mathbf{W}_{m,n}^{\mathsf{H}}$ as precoding matrix that can be computed using (5.4). Normalization is performed to ensure that the average energy per transmitted symbol equals one, by applying the diagonal matrix

$$\mathbf{C}_{m,n} = \left(\left(\mathbf{W}_{m,n}\mathbf{W}_{m,n}^{\mathsf{H}}\right) \odot \mathbf{I}_K\right)^{-\frac{1}{2}} \tag{5.11}$$

leading to the precoded signal

$$\mathbf{t}_{m,n} = \mathbf{C}_{m,n}\mathbf{W}_{m,n}^{\mathsf{H}}\mathbf{s}_{m,n}. \tag{5.12}$$

The channel transfer function (5.10) can be rewritten as

$$\begin{aligned}\mathbf{u}_{m,n} &= \mathbf{H}_{m,n}^{\mathsf{H}}\mathbf{t}_{m,n} + \mathbf{q}_{m,n} \\ &= \underbrace{\mathbf{H}_{m,n}^{\mathsf{H}}\mathbf{C}_{m,n}\mathbf{W}_{m,n}^{\mathsf{H}}}_{\mathbf{G}_{m,n}}\mathbf{s}_{m,n} + \mathbf{q}_{m,n}\end{aligned} \tag{5.13}$$

where $\mathbf{G}_{m,n} \in \mathbb{C}^{K\times K}$ is referred to as the *equivalent* downlink channel for the RE (m,n). Each user k receives its signal $u_{m,n}^{(k)}$ and the corresponding channel, i.e., the k^{th} row of $\mathbf{G}_{m,n}$, is denoted by $\mathbf{g}_{m,n}^{(k)\,\mathsf{T}} \in \mathbb{C}^K$. Finally, the equivalent channel experienced by user k for the entire RG is denoted by $\mathbf{G}^{(k)} \in \mathbb{C}^{2M\times N\times K}$.

Channel estimation and equalization

To enable estimation of the equivalent downlink channel by the users, pilot signals are transmitted by the BS using the same pilot patterns as in the uplink (Fig. 5.1). Each user k estimates its equivalent channel $\widehat{\mathbf{G}}^{(k)} \in \mathbb{C}^{2M\times N\times K}$, where for a given RE (m,n), the element $\hat{g}_{m,n,k}^{(k)}$ corresponds to the main channel coefficient, whereas the elements $\hat{g}_{m,n,i}^{(k)}, i \neq k$ correspond to the interference channel coefficients. As in the uplink, LMMSE estimation, followed by spectral and possibly temporal interpolation, is used, but it is assumed that the elements of $\hat{\mathbf{g}}_{m,n}^{(k)}$ are uncorrelated. Therefore, channel estimation is performed independently for the main channel and each interference channel, enabling easy scalability to any number of interferers. The covariance matrices used to estimate the main channel and one of the interfering channels of a given user are denoted by $\mathbf{\Omega} \in \mathbb{C}^{|\mathcal{P}_M|\cdot|\mathcal{P}_N|\times|\mathcal{P}_M|\cdot|\mathcal{P}_N|}$ and by $\mathbf{\Psi} \in \mathbb{C}^{|\mathcal{P}_M|\cdot|\mathcal{P}_N|\times|\mathcal{P}_M|\cdot|\mathcal{P}_N|}$, respectively, and are equal for all users and interfering channels. The

tensor of the equivalent channel estimation error for user k is denoted by $\widetilde{\mathbf{G}}^{(k)} \in \mathbb{C}^{2M\times N\times K}$, and is such that $\mathbf{G}^{(k)} = \widehat{\mathbf{G}}^{(k)} + \widetilde{\mathbf{G}}^{(k)}$. The estimation error variances for the main and the i^{th} interfering channel of user k are respectively denoted by $v^{(k)}_{m,n,k} \coloneqq \mathbb{E}\left[\left|\tilde{g}^{(k)}_{m,n,k}\right|^2\right]$ and $v^{(k)}_{m,n,i} \coloneqq \mathbb{E}\left[\left|\tilde{g}^{(k)}_{m,n,i}\right|^2\right]$. Similarly, we denote by $\mathbf{V}^{(k)} \in \mathbb{R}^{N,\times M\times K}$ the tensor of estimated error variances for user k, as shown in Fig. 5.5. An estimation of the transmitted unprecoded symbol for user k is computed by equalizing the received signal as follows

$$\hat{s}^{(k)}_{m,n} = \frac{u^{(k)}_{m,n}}{\hat{g}^{(k)}_{m,n,k}}. \tag{5.14}$$

Demapping

The post-equalization downlink channel can be seen as $M \times N \times K$ parallel additive noise channels. More precisely, for a user $k \in \{1,\dots,K\}$ and RE (m,n),

$$\hat{s}^{(k)}_{m,n} = s_{m,n,k} + \underbrace{\frac{\tilde{g}^{(k)}_{m,n,k} s_{m,n,k} + \mathbf{g}^{(k)\ \mathsf{T}}_{m,n,-k} \mathbf{s}_{m,n,-k} + q_{m,n,k}}{\hat{g}^{(k)}_{m,n,k}}}_{\xi_{m,n,k}} \tag{5.15}$$

where $\xi_{m,n,k}$ comprises the channel noise and interference and has variance

$$\begin{aligned}\tau^2_{m,n,k} &= \mathbb{E}\left[\xi^*_{m,n,k}\xi_{m,n,k}\right] \\ &= \frac{v^{(k)}_{m,n,k} + \hat{\mathbf{g}}^{(k)\ \mathsf{H}}_{m,n,-k}\, \hat{\mathbf{g}}^{(k)}_{m,n,-k} + \sum_{i=1,i\neq k}^{K} v^{(k)}_{m,n,i} + \sigma^2}{\left|\hat{g}^{(k)}_{m,n,k}\right|^2}.\end{aligned} \tag{5.16}$$

Assuming $\xi_{m,n,k}$ is Gaussian distributed[2], the LLR for the q^{th} bit transmitted to user k on RE (m,n) is given by

$$\text{LLR}^{\text{DL}}_{m,n,k}(q) = \ln\left(\frac{\sum_{c\in\mathcal{C}_{q,1}} \exp\left(-\frac{1}{\tau^2_{m,n,k}}\left|\hat{s}_{m,n,k} - c\right|^2\right)}{\sum_{c\in\mathcal{C}_{q,0}} \exp\left(-\frac{1}{\tau^2_{m,n,k}}\left|\hat{s}_{m,n,k} - c\right|^2\right)}\right). \tag{5.17}$$

5.2.4 Estimation of the Required Statistics

The baselines described above require the knowledge of the covariance matrices $\mathbf{\Sigma}$, $\mathbf{\Omega}$, and $\mathbf{\Psi}$ which provide the spatial, time, and spectral correlations between the REs carrying pilots. These matrices can be set based on models or can be empirically estimated by constructing large datasets of uplink, downlink, and interfering pilot signals, as was done in this Chapter. The channel estimation error covariances $\mathbf{E}_{m,n}$, defined in (5.3), also need to be estimated to

[2]Similarly to the uplink scenario, this is not true in general.

compute both the uplink equalization matrices $\mathbf{W}_{m,n}$ and the downlink precoding matrices $\mathbf{W}_{m,n}^{\mathsf{H}}$. Focusing on REs carrying pilots, the estimation error covariance for a user k is

$$\begin{aligned}\mathbf{E}_{\mathcal{P}^{(k)}}^{(k)} &= \mathbb{E}\left[\operatorname{vec}\left(\widetilde{\mathbf{H}}_{\mathcal{P}^{(k)}}^{(k)}\right)\operatorname{vec}\left(\widetilde{\mathbf{H}}_{\mathcal{P}^{(k)}}^{(k)}\right)^{\mathsf{H}}\right] \\ &= \mathbf{\Sigma} - \mathbf{\Sigma}\left(\mathbf{\Sigma} + \sigma^2\right)^{-1}\mathbf{\Sigma}\end{aligned} \tag{5.18}$$

where $\widetilde{\mathbf{H}}_{\mathcal{P}^{(k)}}^{(k)}$ is the channel estimation error at REs carrying pilots. However, we are only interested in the *spatial* channel estimation error correlations, whereas (5.18) provides the correlations of channel estimation errors between all the receive antennas, subcarriers, and symbols. For a single pilot position $(m,n) \in \mathcal{P}^{(k)}$, this spatial correlation matrix is defined by

$$\mathbf{E}_{(m,n)\in\mathcal{P}^{(k)}}^{(k)} = \mathbb{E}\left[\left(\widetilde{\mathbf{h}}_{(m,n)\in\mathcal{P}^{(k)}}^{(k)}\right)\left(\widetilde{\mathbf{h}}_{(m,n)\in\mathcal{P}^{(k)}}^{(k)}\right)^{\mathsf{H}}\right] \in \mathbb{C}^{L\times L} \tag{5.19}$$

and can be extracted from $\mathbf{E}_{\mathcal{P}^{(k)}}^{(k)}$. To estimate $\mathbf{E}_{m,n}$ for REs carrying data, a nearest-pilot approach is used, which sets the value $\mathbf{E}_{m,n}$ for a RE carrying a data signal to the one of the nearest RE carrying a pilot signal. The so-obtained estimation is denoted by $\widehat{\mathbf{E}}_{m,n}^{(k)}$ for a RE (m,n). The overall spatial estimation error covariance matrix for any RE (m,n) is obtained by summing the estimations for all users:

$$\widehat{\mathbf{E}}_{m,n} = \sum_{k=1}^{K} \widehat{\mathbf{E}}_{m,n}^{(k)} \quad \in \mathbb{C}^{L\times L}. \tag{5.20}$$

The uplink channel and error covariance estimations are depicted in Fig. 5.3.

In the downlink, the estimation error variances $v_{m,n,k}^{(k)}$ and $v_{m,n,i}^{(k)}, i \neq k$ for user k are estimated following a similar procedure, but with only one receive antenna and using the downlink covariances matrices $\mathbf{\Omega}$ and $\mathbf{\Psi}$. The resulting quantity is denoted by $\widehat{\mathbf{V}}^{(k)}$, as shown in Fig. 5.5.

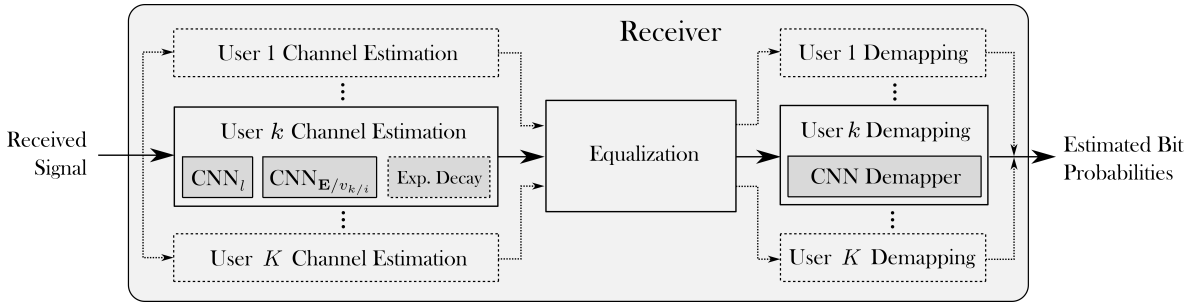


Figure 5.6: DL-enhanced receiver architecture. The dotted elements are only present in the uplink, where the BS jointly processes all users. The dark gray elements are trainable components.

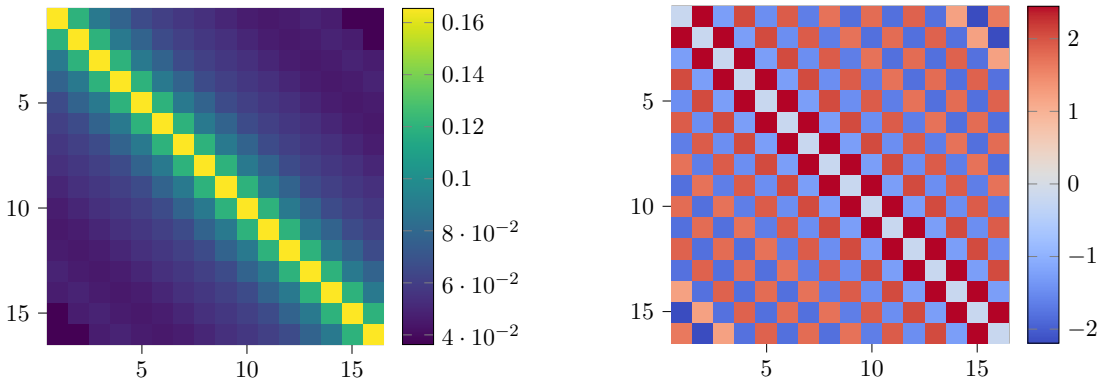
5.3 DL-enhanced Receiver Architecture

The baselines presented in the previous section have several limitations. Especially, the NIRE approximation leads to high channel estimation errors for REs that are far from pilots. Similarly, the grouped equalization can be inaccurate at those REs. This section details the architecture of a receiver that builds on the presented baseline and uses multiple CNNs to improve its performance.

5.3.1 Receiver Training

The DL-enhanced receiver architecture is shown in Fig. 5.6, where the trainable components are represented in dark gray. In the downlink, each user k only performs the channel estimation, equalization, and demapping of its own signal, and the corresponding components are illustrated with continuous outlines. However, in the uplink, the BS processes all users in parallel, and the additional components are delimited with dotted lines. Although the internal processing of the trainable components might not be interpretable, using multiple DL-based blocks to perform precise and relatively simple signal processing tasks allows to precisely control which parts of the receiver are enhanced. Moreover, this approach makes the output of each DL components easier to interpret, as discussed in Section 5.4.3 for the error statistics $\widehat{\mathbf{E}}$.

Scalability is achieved by using different copies of the same DL components for every user, where all copies share the same set of trainable parameters. We propose to jointly optimize all these components based only on the estimated bit probabilities, and not by training each of them individually. This approach is practical as it does not assume knowledge of the channel coefficients at training, that can only be estimated through extensive measurement campaigns for practical channels. Let's denote by $\boldsymbol{\theta}$ the set of trainable parameters of the DL-enhanced receiver, and by $b_{m,n,k,q}$ the transmitted bit (m, n, k, q). In the uplink, those parameters are



(a) Amplitude (b) Phase

Figure 5.7: Example of amplitude and phase for $\mathbf{E}_{m,n}^{(k)}$.

optimized to minimize the total BCE:

$$\mathcal{L} \triangleq -\sum_{k=1}^{K} \sum_{(m,n)\in\mathcal{D}} \sum_{q=0}^{Q-1} \mathbb{E}_{b,\mathbf{Y}} \left[\log_2 \left(\widehat{P}_{\boldsymbol{\theta}} \left(b_{m,n,k,q} | \mathbf{Y} \right) \right) \right] \tag{5.21}$$

where $\mathcal{D}$ denotes the set of REs carrying data and $\widehat{P}_{\boldsymbol{\theta}}(\cdot|\mathbf{Y})$ is the receiver estimate of the posterior distribution on the bits given $\mathbf{Y}$. In the downlink, the receiver parameters are optimized in a similar manner, except that the signal received by the users is $\mathbf{R}$. The expectation in (5.21) is estimated through Monte Carlo sampling using batches of B_S samples:

$$\mathcal{L} \approx -\frac{1}{B_S} \sum_{i=1}^{B_S} \sum_{k=1}^{K} \sum_{(m,n)\in\mathcal{D}} \sum_{q=0}^{Q-1} \left(\log_2 \left(\widehat{P}_{\boldsymbol{\theta}} \left(b_{m,n,k,q}^{[i]} | \mathbf{Y}^{[i]} \right) \right) \right) \tag{5.22}$$

where the superscript $[i]$ refers to the s^{th} sample in the batch. Following a derivation similar to (2.91), the loss (5.21) can be redefined as

$$\mathcal{L} = \sum_{k=1}^{K} \left(\text{Card}(\mathcal{D}) Q - C_k \right) \tag{5.23}$$

where $\text{Card}(\mathcal{D})$ is the number of REs carrying data, $\text{Card}(\mathcal{D})Q$ is the total number of bits transmitted by one user, and C_k is an achievable rate for user k:

$$\begin{aligned} C_k = &\sum_{(m,n)\in\mathcal{D}} \sum_{q=0}^{Q-1} I \left(b_{m,n,k,q}; \mathbf{Y} \right) \\ &- \sum_{(m,n)\in\mathcal{D}} \sum_{q=0}^{Q-1} \mathbb{E}_{\mathbf{Y}} \left[D_{KL} \left(P \left(b_{m,n,k,q} | \mathbf{Y} \right) || \widehat{P}_{\boldsymbol{\theta}} \left(b_{m,n,k,q} | \mathbf{Y} \right) \right) \right]. \end{aligned} \tag{5.24}$$

Minimizing $\mathcal{L}$ therefore maximizes C_k, which directly translates to improved BER performances.

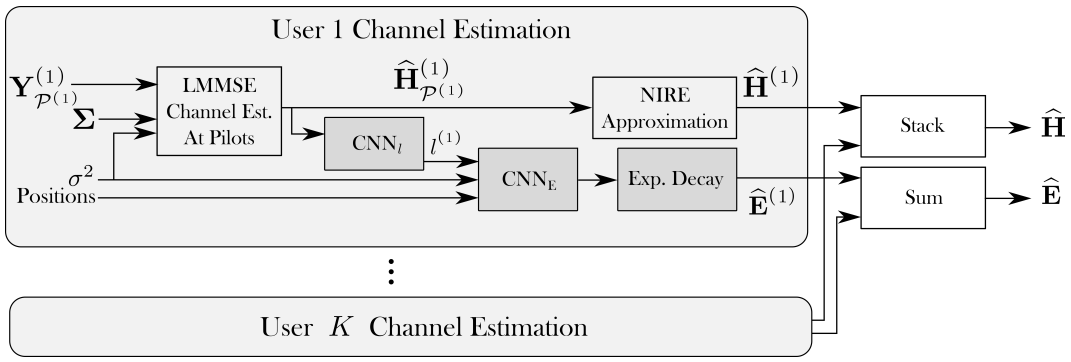


Figure 5.8: DL-enhanced uplink channel estimation.

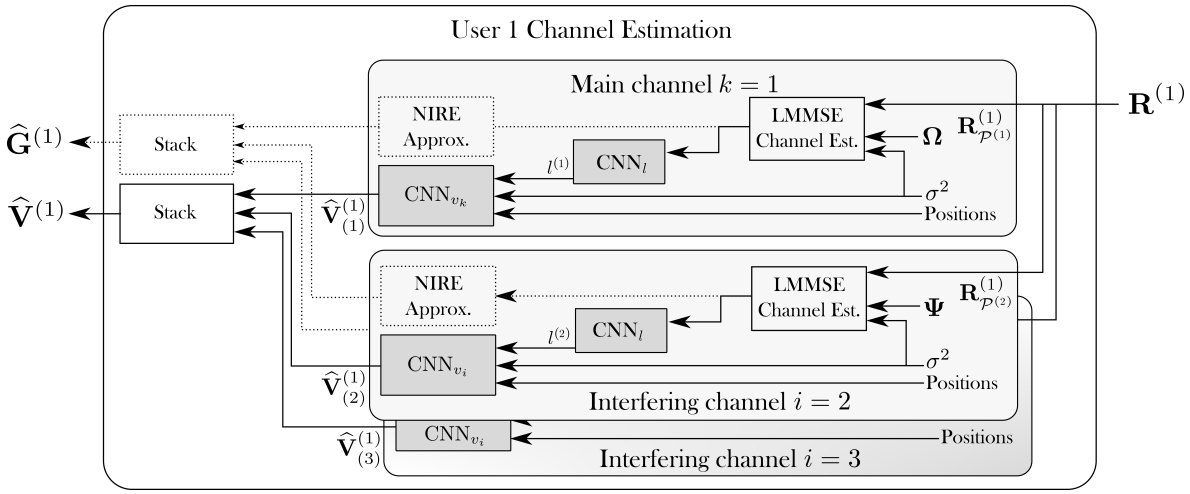


Figure 5.9: Detailed view of the downlink *Channel Estimation* component of user 1 out of $K = 3$, as depicted in Fig 5.5.

5.3.2 DL-enhanced Channel Estimator

As seen in Section 5.2.4, the channel estimation error statistics can only be obtained for REs carrying pilots. However, the estimation accuracy decreases as we move away from them. In the following, we present CNNs that estimate the channel estimation error covariance matrices in the uplink and the estimation error variances in the downlink.

Uplink scenario

The DL-enhanced uplink channel estimation architecture is depicted in Fig. 5.8. In this scenario, the spatial channel estimation error covariance matrices $\mathbf{E}_{m,n}$ is needed to compute both the equalized symbols (5.4) and the uplink post-equalization noise variance (5.8). An example of the amplitude and phase of an estimate of a covariance matrix $\widehat{\mathbf{E}}_{1,1}^{(1)}$ is shown in Fig. 5.7 for a uniform linear array (ULA) of antennas at the BS. One can see that the amplitude of the coefficients of $\widehat{\mathbf{E}}_{1,1}^{(1)}$ decays rapidly when moving away from the diagonal. The phase, on the other hand, exhibits a more surprising pattern, with a phase difference

of roughly π between two adjacent antennas. To predict every element of $\widehat{\mathbf{E}}^{(k)}$ for user k, a naively designed CNN would need to output MNL^2 complex parameters. This would be of prohibitive complexity for any large number of subcarriers, symbols, or receiving antennas. For this reason, we propose to approximate every element (x, y) of $\widehat{\mathbf{E}}^{(k)}_{m,n}$ with a complex power decay model:

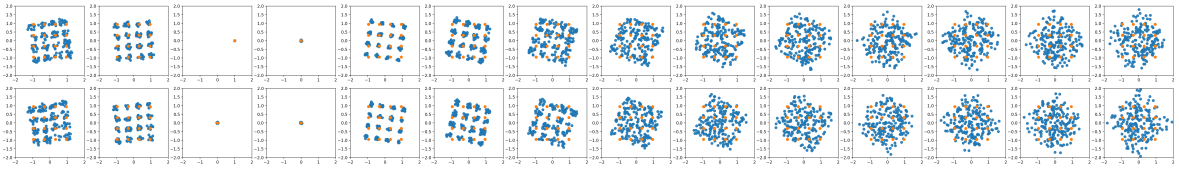
$$\hat{e}^{(k)}_{m,n,a,b} = \alpha_{m,n}\beta^{|b-a|}_{m,n} \exp\left(j\gamma(b-a)\right) \tag{5.25}$$

where $b-a$ is the horizontal position difference between that element and the diagonal, and $\alpha_{m,n}$, $\beta_{m,n}$, and γ are parameters of this model. For a planar array, one could use such a model for each dimension, and take their Kronecker product to obtain the spatial channel estimation error covariance matrices. A constant phase offset between two adjacent REs is assumed, which matches our experimental observations. The parameters $\alpha_{m,n}$ and $\beta_{m,n}$ respectively control the scale and the decay of the model, and depend on the RE (m, n).

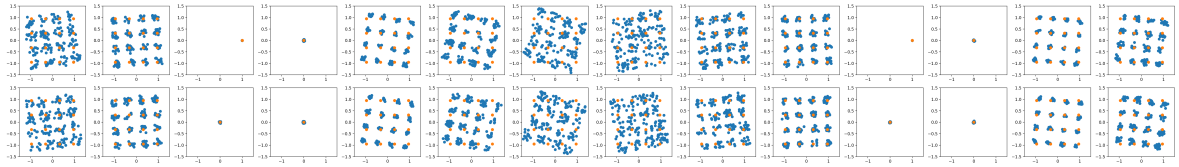
To estimate those two parameters for each RE, we use a CNN, denoted by $\text{CNN}_{\mathbf{E}}$, which takes four inputs, each of size $M \times N$, for a total input dimension of $M \times N \times 4$. $\text{CNN}_{\mathbf{E}}$ outputs $\alpha_{m,n}$ and $\beta_{m,n}$ for every REs, resulting in an output dimension of $M \times N \times 2$. The first two inputs provide the location of every REs in the RG. More precisely, the first input matrix has all columns equal to $[-\frac{M}{2}, \dots, -1, 1, \dots, \frac{M}{2}]^\mathsf{T}$, whereas the second one has all rows equal to $[-\frac{N}{2}, \dots, -1, 1, \dots, \frac{N}{2}]$. The third input provides the SNR of the transmission and is given as a matrix $\text{SNR} \cdot \mathbb{1}_{M\times N}$. Finally, the fourth input is a feature $l^{(k)} \in \mathbb{R}$ provided by another CNN, denoted by CNN_l, which was designed with the intuition to predict the time-variability of the channel experienced by user k. To do so, CNN_l uses the channel estimates at REs carrying pilots to estimate the Doppler and delay spread. Although we cannot be certain that CNN_l effectively learns to extract such information, the evaluations presented in Section 5.4.3 tend to support this hypothesis. CNN_l takes an input of dimension $|\mathcal{P}_M| \times |\mathcal{P}_N| \times 2L$, which corresponds to the stacking of the real and imaginary parts of $\widehat{\mathbf{H}}^{(k)}_{\mathcal{P}^{(k)}}$ along the last dimension. It outputs the scalar $l^{(k)}$, which is fed to $\text{CNN}_{\mathbf{E}}$ as the matrix $l^{(k)} \cdot \mathbb{1}_{M\times N}$.

Downlink scenario

In the downlink, the equivalent channel estimation error variances $\mathbf{V}^{(k)}$ are needed to compute $\tau^2_{m,n,k}$ in (5.16). To estimate those variances, we take inspiration from the architecture presented in Section 5.2 which uses two different downlink covariance matrices $\mathbf{\Omega}$ and $\mathbf{\Psi}$ to estimate the error variances of the main and interfering channels. Similarly, we use two separate CNNs, denoted by CNN_{v_k} and by CNN_{v_i}, to respectively predict the estimation error variances of the main channel $\hat{v}^{(k)}_{m,n,k}$ and of an interfering channel $\hat{v}^{(k)}_{m,n,i}, i \neq k$. Both CNNs take the same inputs as $\text{CNN}_{\mathbf{E}}$ but their outputs are of dimension $M \times N$ as variances are predicted for all REs (m, n). To preserve the scalability of the conventional architecture, $K-1$ copies of CNN_{v_i} are used to estimate the error variances $\hat{v}^{(k)}_{m,n,i}$ of all $K-1$ interferers. The downlink channel estimation is schematically shown in Fig. 5.9, where $\widehat{\mathbf{V}}^{(k)}_{(a)} = \{\hat{v}^{(k)}_{m,n,a}\}_{(m,n)\in\{0,\dots,M\}\times\{0,\dots,N\}}$ denotes the estimation error variances seen by user k on its main or interfering channel a.



(a) 1P pilot pattern.



(b) 2P pilot pattern.

Figure 5.10: Mismatch between the transmitted signals (orange) and the equalized received ones (blue) for a single user out of four and using 16-QAM modulation.

5.3.3 DL-enhanced Demapper

A consequence of unperfect channel estimation and equalization is channel aging, which leads to residual distortion on the equalized signals, as illustrated in Fig. 5.10. The signals transmitted by a single user out of four are represented in orange, while the corresponding equalized received signals are shown in blue, assuming spectral channel interpolation only. This figure has been obtained by sending a large batch of signals using a fixed realization of a fast-varying channel, and only displays the first two subcarriers of the uplink slot. Moreover, an infinite SNR is assumed so that only the effects of channel aging and interference are visible. One can see that the equalized symbols suffer from little distortion and interference at REs close to the pilots, but these unwanted effects become increasingly stronger at REs that are away from them.

A traditional demapper, as presented in Section 5.2.2, operates independently on each RE and therefore only sees one equalized symbol at a time. In contrast, we propose to use a CNN, called CNN_{Dmp}, to perform a joint demapping of the entire RG. By jointly processing all equalized symbols, the CNN can estimate and correct the effects of channel aging to compute

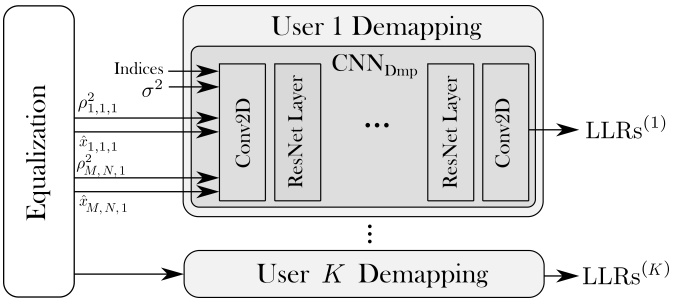


Figure 5.11: CNN-based uplink demapper.

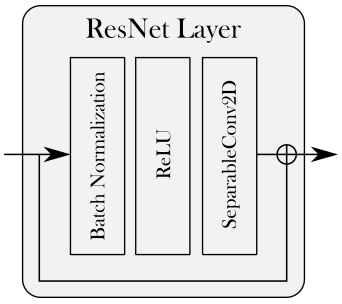


Figure 5.12: ResNet layer.

	$\text{CNN}_{\mathbf{E}}/\text{CNN}_{v_{k,i}}$				CNN_l			CNN_{Dmp}		
Input size	$M \times K \times 4$				$N_{P_f} \times N_{P_t} \times 2L$			$M \times K \times 6$		
Parameters	filt.	kern.	dilat.	act.	filt.	kern.	dilat.	filt.	kern.	dilat.
Conv2D	32	(5,3)	(1,1)	ReLU	32	(1,1)	(1,1)	128	(1,1)	(1,1)
ResNet Layer	-				32	(3,2)	(1,1)	128	(3,3)	(1,1)
ResNet Layer	-				32	(5,2)	(2,1)	128	(5,3)	(2,1)
ResNet Layer	-				32	(7,2)	(3,1)	128	(7,3)	(3,2)
ResNet Layer	-				32	(5,2)	(2,1)	128	(9,3)	(4,3)
ResNet Layer	-				32	(3,2)	(1,1)	128	(7,3)	(3,2)
ResNet Layer	-				-			128	(5,3)	(2,1)
ResNet Layer	-				-			128	(3,3)	(1,1)
Conv2D	32	(5,3)	(1,1)	ReLU	1	(3,2)	(1,1)	Q	(1,1)	(1,1)
Conv2D	2 / 1	(1,1)	(1,1)	Sigm.	-			-		
Output Layer	-				Dense, units = 1			-		
Output size	$M \times K \times 2/M \times K \times 1$				1			$M \times K \times Q$		

Table 5.1: Parameters of the different CNNs.

better LLRs. The input of CNN_{Dmp} is of dimension $M \times N \times 6$ and carries the subcarriers and symbol indices for each RE, the SNR, the real and imaginary parts of the equalized symbols, and the post-equalization channel noise variances. The output of CNN_{Dmp} has dimensions $M \times N \times Q$ and corresponds to the predicted $\text{LLRs}^{(k)}$ over the RG for a user k. As with a conventional receiver, the demapping is performed independently for each user to make the architecture easily scalable. The CNN_{Dmp} demapper is shown in Fig. 5.11, which depicts the uplink demapping process.

5.3.4 CNN Architectures

All CNNs presented above share the same building blocks: convolutional 2D layers, dense layers, and custom ResNet layers. The custom ResNet layers, inspired by [84], consist of a batch normalization layer, a ReLU, a 2D separable convolutional layer, and finally the

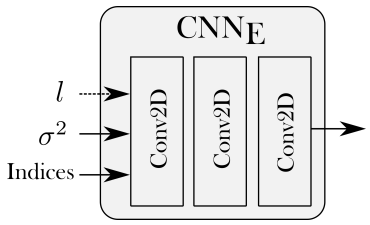


Figure 5.13: Architecture of CNN$_{\mathbf{E}}$.

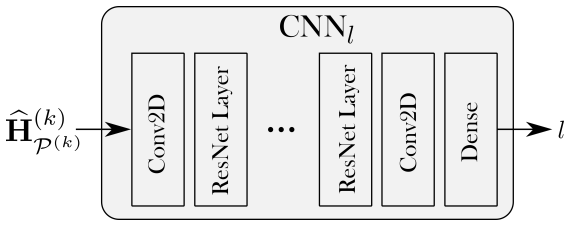


Figure 5.14: Architecture of CNN$_l$.

addition of the input, as depicted in Fig. 5.12. Separable convolutions are less computationally expensive while maintaining similar performance as regular convolutional layers [31]. CNN$_{\mathbf{E}}$, CNN$_{v_k}$, and CNN$_{v_i}$ are all made of three 2D convolutional layers, as shown in Fig. 5.13 for CNN$_{\mathbf{E}}$. The first two layers are followed by a ReLU activation function, while the last layer is followed by a sigmoid activation function. CNN$_{\text{Dmp}}$ and CNN$_f$ also share a similar architecture, depicted in Fig. 5.11 and Fig 5.14, respectively. Both are composed of a 2D convolutional layer, followed by multiple ResNet layers and a 2D convolutional layer. CNN$_f$ outputs a single scalar, which is ensured by using a dense layer with a single unit and no activation function as output layer. Inspired by [81], we used increasing followed by decreasing kernel sizes and dilation rates to increase the receptive field of the CNN. All convolutional and separable convolutional layers use zero-padding so that the output dimensions matches the input dimensions. Details of the architectures of all CNNs are given in Table 5.1.

Parameters	Symbol (if any)	Value
Number of users	K	4 (2 in high speed downlink)
Number of antennas at the BS	L	16
Number of subcarrier	N	72 = 6 resource blocks
Number of OFDM symbols	M	14 (uplink) + 14 (downlink)
Bit per channel use	Q	4 bit (uplink), 2 bit(downlink)
Center frequency	-	3.5 GHz
Subcarrier spacing	-	15 kHz
Scenario	-	3GPP 38.901 UMi NLOS
Code length	-	1296 bit
Code rate	η	$\frac{1}{2}$ (uplink), $\frac{1}{3}$ (downlink)
Learning rate	-	10^{-3}
Batch size	B_S	27 RGs

Table 5.2: Training and evaluation parameters.

5.4 Evaluations

In this section, the proposed DL-enhanced receiver is evaluated and compared against two baselines: the one presented in Section 5.2 as well as a perfect CSI baseline that will be detailed later on. The training and evaluation setups are first introduced, followed by evaluations of the uplink and downlink performance for the 1P and 2P pilot patterns from Fig. 5.1.

5.4.1 Training and Evaluation Setup

For realistic training and evaluation, the channel realizations were generated with QuaDRiGa version 2.0.0 [85]. It has been experimentally observed in [81] that a receiver trained on certain scenarios was able to generalize to other channel models. For this reason, we only focus on the 3GPP non-line of sight (NLOS) UMi scenario [94]. The number of users was set to $K = 4$, except for the downlink at high speeds where it was reduced to $K = 2$, and the number of antennas at the BS was set to $L = 16$. All users were randomly placed within a 120° cell sector, with a minimum distance of 15 m and a maximum distance of 150 m from the BS. The user and BS heights were respectively set to 1.5 m and 10 m. The RGs were composed of six resource blocks for a total of $N = 72$ subarriers, with a center frequency of 3.5 GHz and a subcarrier spacing of 15 kHz. Both the uplink and downlink slots contained $M = 14$ OFDM symbols. A Gray-labeled QAM was used with $Q = 4$ bits per channel use on the uplink and $Q = 2$ bits per channel use on the downlink. The receivers were trained and evaluated on users

moving at independent random speeds. Three ranges of low speeds were considered with the 1P pilot pattern: 0 to 15 km h^{-1}, 15 to 30 km h^{-1}, and 30 to 45 km h^{-1}. Similarly, three high speed ranges were considered with the 2P pilot pattern: 50 to 70 km h^{-1}, 80 to 100 km h^{-1}, and 110 to 130 km h^{-1}. We noticed that CNN$_f$ was not able to extract useful information when using the 1P pattern, and therefore was not used in the corresponding trainings and evaluations. For equalization, the grouped-LMMSE equalizer operated on groups of 2×7 REs, following the segmentation delimited by the thick black line of the 2P pattern in Fig. 5.1c. The training and evaluation parameters are given in Table 5.2.

Separate training sets were constructed for the 1P and 2P pilot pattern, both made of channel realizations corresponding to 1000 RGs of each respective user speed range, for a total of 3000 RGs per training dataset. Evaluations were performed separately for each user speed range with datasets containing 3000 RGs. A standard IEEE 802.11n low-density parity-check (LDPC) code of length 1296 bit [95] was used, with code rates of $\eta = \frac{1}{2}$ and $\eta = \frac{1}{3}$ for uplink and downlink transmissions, respectively. Decoding was done with 40 iterations using a conventional belief-propagation decoder. To satisfy the perfect power assumption in (5.1), each RG was normalized to have an average energy of one per antenna and per user, i.e.,

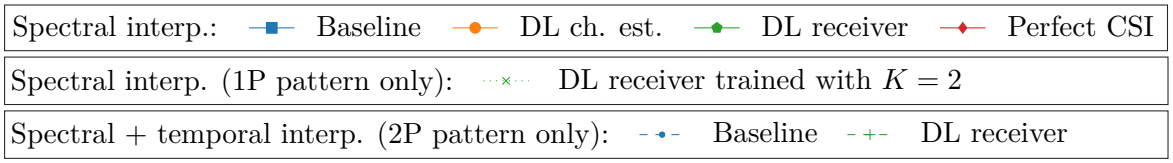
$$\sum_{m=1}^{2M} \sum_{n=1}^{N} ||\mathbf{h}_{m,m}^{(k)}||_2^2 = 2MNL. \tag{5.26}$$

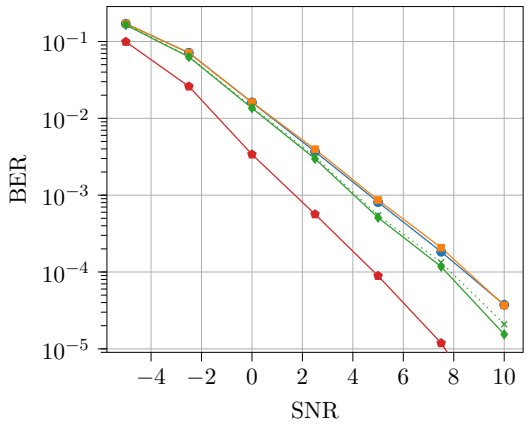
Each training was carried out using batches of size $B_S = 27$ so that the total number of bits transmitted for each user was a multiple of the code length. The trainable parameters $\boldsymbol{\theta}$ were all initialized randomly except for γ in (5.25) that was initialized with π. Training was done through SGD using the Adam [30] optimizer and a learning rate of 10^{-3}. The best performing random initialization out of ten was selected.

5.4.2 Uplink Simulation Results

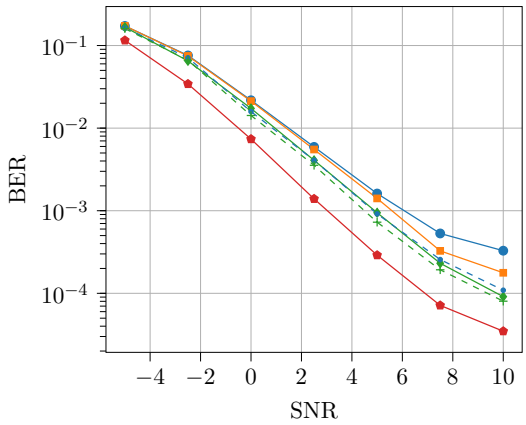
Different schemes were benchmarked in the uplink simulations. The first one is the uplink baseline presented in Sections 5.2.2 and 5.2.4. The second one, named 'DL channel estimator', uses the DL-enhanced channel estimator presented in 5.3.2 but is trained and tested with a standard demapper. The third one is the DL-enhanced receiver, leveraging both the enhanced channel estimator and demapper. We refer to it as 'DL receiver'. Evaluating the DL channel estimator separately allows us to better understand the role both components play in the observed gains. An ideal baseline with perfect knowledge of the channel at the REs carrying pilots and of $\mathbf{E}$ is also considered, and referred to as 'Perfect CSI'. All schemes used spectral interpolation followed by NIRE approximation for channel estimation. Additional simulations were conducted for the 2P pilot pattern using spectral and temporal interpolation for both the baseline and the DL receiver. Finally, a DL receiver trained with only $K = 2$ users was also evaluated to study the scalability of the system.

Simulation results for the 1P pattern are shown in the first column of Fig. 5.15 for the three different speed ranges. At speeds ranging from 0 to 15 km h^{-1}, one can see that the DL

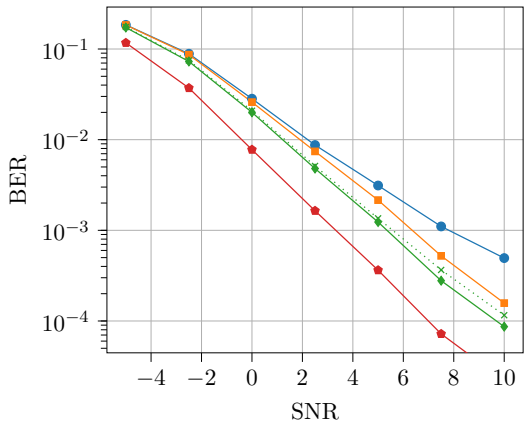




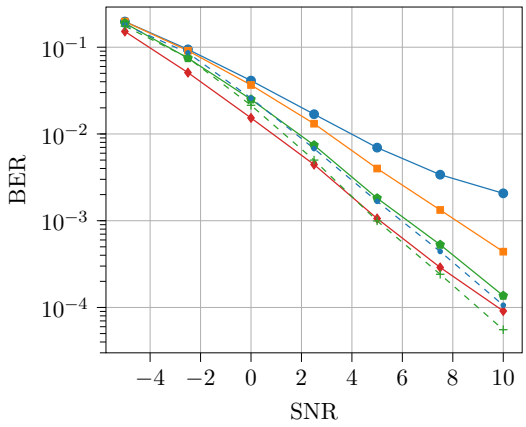
(a) 1P pilot pattern at 0 to 5 km h^{-1}.



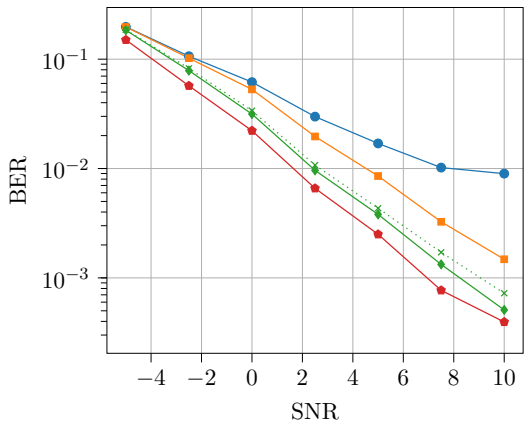
(b) 2P pilot pattern at 40 to 60 km h^{-1}.



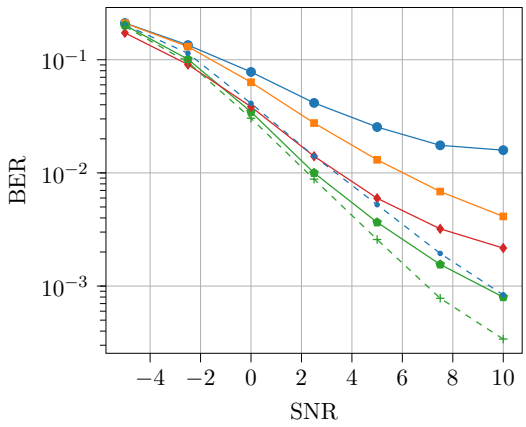
(c) 1P pilot pattern at 10 to 20 km h^{-1}.



(d) 2P pilot pattern at 70 to 90 km h^{-1}.



(e) 1P pilot pattern at 25 to 35 km h^{-1}.



(f) 2P pilot pattern at 110 to 130 km h^{-1}.

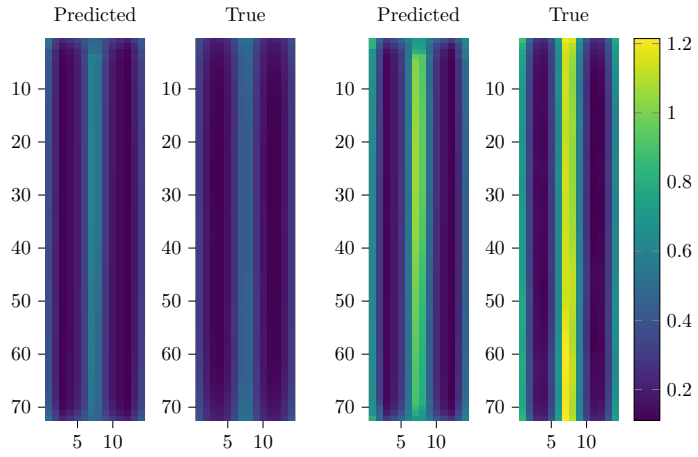
Figure 5.15: Uplink BER achieved by the different receivers with the 1P and 2P pilot patterns.

channel estimator does not bring any improvements, whereas the DL receiver achieves a 1 dB gain at a coded BER of 10^{-3}. The benefit of having a better estimation of $\mathbf{E}$ becomes more significant as the speed increases. At the highest speed range (Fig. 5.15e), the DL receiver enables gains of 3 dB over the baseline at a coded BER of 10^{-2}. It can be observed that the DL receiver trained with only two users nearly matches the performance of the one trained with four users on all considered speeds, demonstrating the scalability of the proposed scheme with respect to the number of users.

Results for the 2P pilot pattern are shown in the second column of Fig. 5.15. The gains provided by the DL channel estimator alone and by the entire DL receiver follow the same trend as with the 1P pattern, with moderate gains at 50-70 $\mathrm{km\,h^{-1}}$, but significant improvements at 110-130 $\mathrm{km\,h^{-1}}$. More precisely, the DL receiver provides a 1 dB gain over the baseline at a coded BER of 10^{-3} in the 50-70 $\mathrm{km\,h^{-1}}$ range, and is the only scheme that achieves a coded BER of 10^{-3} for the highest speeds with spectral interpolation only. Indeed, at high speeds, the learned demapper is still able to mitigate the effects of channel aging, whereas even the perfect CSI baseline suffers from strongly distorted equalized signals. Using both spectral and temporal interpolations reduces the gains provided by the DL receiver, which can be explained by the better channel estimates leading to less channel aging, but they still amount to a 2.2 dB gap at a BER of 10^{-3} for the highest speeds. We have also experimentally verified that a DL receiver trained with only $K = 2$ users was able to closely match the performance of the one trained with $K = 4$, but decided not to include the corresponding curves for clarity reasons. Overall, one can see that only the combination of a CNN-based estimation of the channel estimation error statistics and CNN-based demapper enables gains for both pilot patterns and all speed ranges.

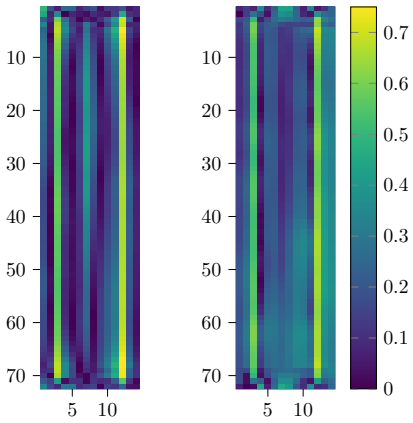
5.4.3 Visualizing the Channel Estimation Error Statistics

In order to get insights into the DL channel estimator abilities, we visualize the channel estimation error covariance matrices $\mathbf{E}$ for different user speeds. Two batches of uplink signals were sent, with users respectively moving at 60 $\mathrm{km\,h^{-1}}$ and 120 $\mathrm{km\,h^{-1}}$. All batches comprised 90 RGs, and used the 2P pilot pattern with an SNR of 5 dB. The Frobenius norms $||\mathbf{E}_{m,n}||_{\mathrm{F}}$ corresponding to all REs are shown in Fig. 5.16, and the normalized errors $\left|\frac{||\widehat{\mathbf{E}}_{m,n}||_{\mathrm{F}} - ||\mathbf{E}_{m,n}||_{\mathrm{F}}}{||\mathbf{E}_{m,n}||_{\mathrm{F}}}\right|$ are shown in Fig. 5.17. The figures labeled as 'Predicted' refers to the estimations $\widehat{\mathbf{E}}_{m,n}$ produced by the DL channel estimator and averaged over the corresponding batches, using spectral interpolation only. As expected, the Frobenius norms of the REs carrying data strongly depend on their distance to their closest pilot, reflecting the distortions visible in Fig. 5.10b. The figures labeled as 'True' are presented for reference, and are computed by Monte Carlo simulations assuming knowledge of the true channel realizations. One can see that the normalized errors are low on REs carrying data, confirming that $\mathrm{CNN}_{\mathbf{E}}$ is able to learn the channel estimation error statistics from the data during training. The higher errors on REs carrying pilots are due to the loss (5.22) taking solely into account the positions $(m, n) \in \mathcal{D}$, giving no opportunities for $\mathrm{CNN}_{\mathbf{E}}$ to learn the statistics at pilot positions. Experiments



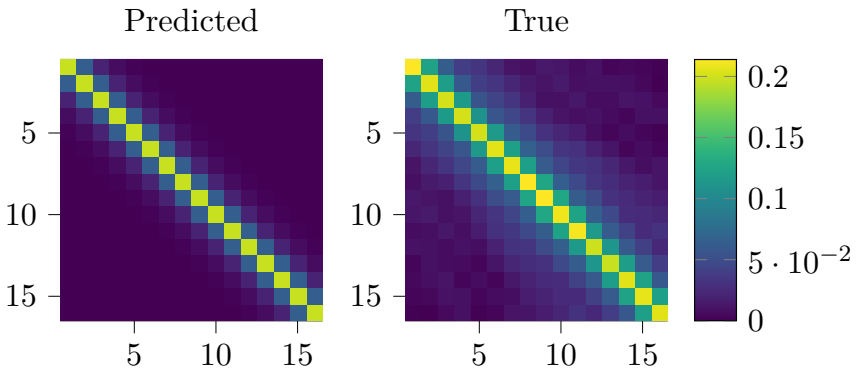
(a) 60 km h^{-1} (b) 120 km h^{-1}

Figure 5.16: Predicted $||\widehat{\mathbf{E}}_{m,n}||_{\mathrm{F}}$ vs true $||\mathbf{E}_{m,n}||_{\mathrm{F}}$ for different user speeds.

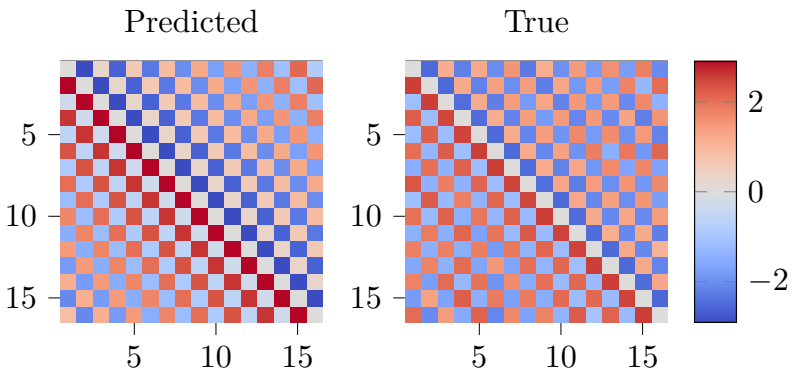


(a) 60 km h^{-1} (b) 120 km h^{-1}

Figure 5.17: Normalized error for different user speeds.



(a) Amplitude



(b) Phase

Figure 5.18: Amplitude and phase of $\widehat{\mathbf{E}}_{7,36}$ and $\mathbf{E}_{7,36}$ for a single realization of the channel at 120 km h^{-1}.

conducted using both spectral and temporal interpolation yielded higher normalized errors, probably because the increased accuracy of the channel estimates results in a more difficult learning of the error statistics. Additionally, CNN_l seems to correctly estimate the time-variability of the channels since the Frobenius norms of the predicted covariances increase with the user speed, matching the behavior of the true covariances.

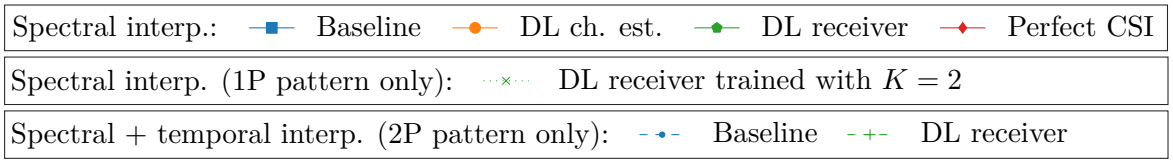
It is also insightful to look at the predicted and true spatial covariance matrix for a single RE. We chose to focus on the RE $(7, 36)$, positioned at the center of the OFDM grid, and on the $120\,\text{km}\,\text{h}^{-1}$ scenario (Fig. 5.16b). The amplitudes and phases of all elements of $\widehat{\mathbf{E}}_{7,36}$ and $\mathbf{E}_{7,36}$ are shown in Fig. 5.18a and 5.18b. One can see that the predicted amplitudes and phases are close to the true ones, thus supporting the proposed power decay model.

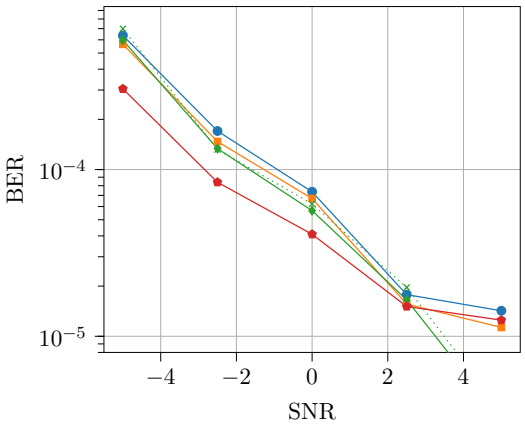
5.4.4 Downlink Simulation Results

Four downlink schemes were evaluated on the considered speed ranges. The first one is the baseline presented in Section 5.2. In the second one, referred to as 'DL channel estimator', each user uses the enhanced channel estimator of Section 5.3.2 and is trained and tested with a standard demapper. The third one is the DL-enhanced receiver using both the enhanced channel estimator and DL demapper for each user, and is referred to as 'DL receiver'. The last scheme has perfect CSI, i.e., all users perfectly know both the channel at REs carrying pilots and the estimation error variances $v^{(k)}_{m,n,k}$ and $v^{(k)}_{m,n,i}$ everywhere. All schemes use the 2P pilot pattern to perform the uplink channel estimation, but are evaluated on both patterns in the downlink. Similarly to the uplink, all schemes leveraged spectral interpolation with NIRE approximation for channel estimation, but additional simulations were performed with the 2P pilot pattern using spectral and temporal interpolation for both the baseline and the DL receiver. A DL receiver trained with only $K = 2$ users was also evaluated on the three slowest speed ranges to study the scalability of the system.

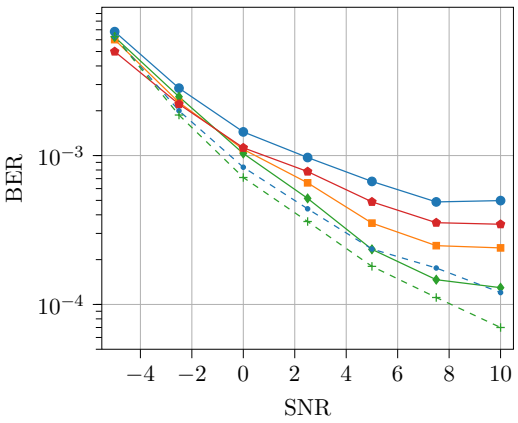
The 1P pilot pattern downlink evaluations are shown in the first row of Fig. 5.19. Between 0 and $15\,\text{km}\,\text{h}^{-1}$, all schemes achieve good results but the DL receivers are the only ones to not saturate at high SNRs. As expected, the gains allowed by the learned channel estimator and demapper increase with the speed, and the DL receiver is the only scheme to reach a BER of approximately 10^{-3} in the 30-45 $\text{km}\,\text{h}^{-1}$ speed range. It can also be observed that the DL receiver trained with only $K = 2$ users is able to closely match the performance of the DL receiver trained with four users. The 2P pilot pattern evaluations are shown in the second row of Fig. 5.19 for higher speeds. In this new setup, the DL receiver outperforms the baseline by 1.1 dB at a BER of 10^{-3} in the speed range 50-70 $\text{km}\,\text{h}^{-1}$, and is the only one with spectral interpolation only to achieve a BER of 10^{-3} in the speed range 70-90 $\text{km}\,\text{h}^{-1}$. Using both spectral and temporal interpolations, the DL receiver still provides significant gains at high speeds, being the only one to achieve a BER of 10^{-3} in the 110-130 $\text{km}\,\text{h}^{-1}$ speed range.

In all downlink scenarios except for the slowest speed range, the DL channel estimator outperforms the perfect CSI baseline. This can be surprising since this baseline uses the exact noise variances $v^{(k)}_{m,n,k}$ and $v^{(k)}_{m,n,i}$ while the DL schemes can only estimate them. We suppose

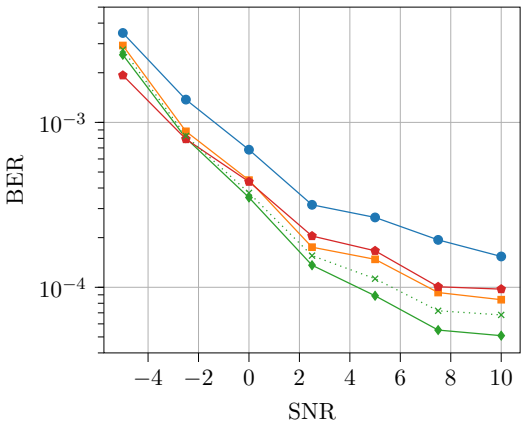




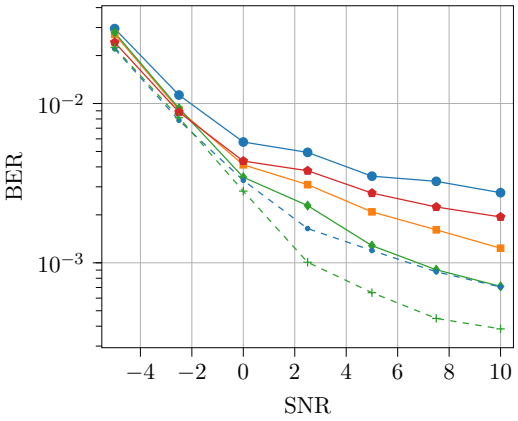
(a) 1P pilot pattern at 0 to 5 km h^{-1}.



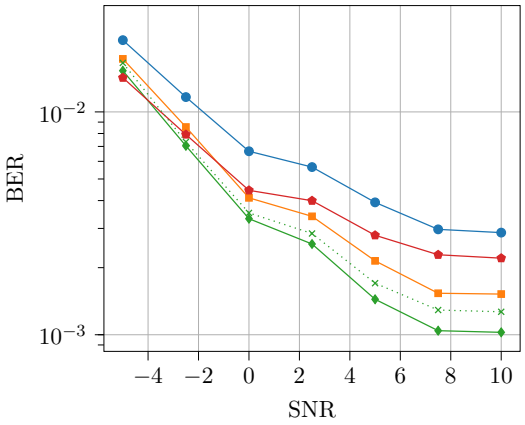
(b) 2P pilot pattern at 40 to 60 km h^{-1}.



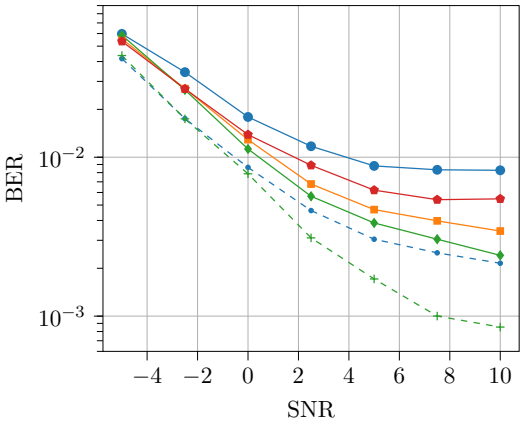
(c) 1P pilot pattern at 10 to 20 km h^{-1}.



(d) 2P pilot pattern at 70 to 90 km h^{-1}.



(e) 1P pilot pattern at 25 to 35 km h^{-1}.



(f) 2P pilot pattern at 110 to 130 km h^{-1}.

Figure 5.19: Downlink BER achieved by the receivers with the 1P and 2P pilot patterns.

that this is because the baselines assume that the post-equalization downlink noise $\xi_{m,n,k}$ in (5.15) is Gaussian distributed and uncorrelated to the transmitted signal, which is typically untrue. The DL channel estimator seems able to learn this model mismatch during training and predict variances that counteract it.

5.5 Concluding Thoughts

A new hybrid strategy to the design of DL-enhaced MU-MIMO receivers was proposed in this chapter. Our architecture builds on top of a traditional MU-MIMO receiver and enhances it with DL components that are trained to maximize the end-to-end performance of the system. More precisely, CNNs are used to improve both the demapping and the computation of the channel estimation error statistics. All components of the proposed architecture are jointly optimized to refine the estimation of the LLRs. This approach does not require any knowledge of the channel for training, is interpretable, and easily scalable to any number of users. Uplink and downlink evaluations were performed with multiple user speeds and two different pilot configurations on 3GPP-compliant channel models. The results reveal that the proposed architecture effectively exploits the OFDM structure to achieve tangible gains at low speeds and significant ones at high speeds compared to a traditional receiver. On the one hand, the enhanced demapper jointly processes all REs of the OFDM grid to counteract the effects of channel aging. On the other hand, we demonstrated that the enhanced channel estimator is able to learn its error statistics during training. In order to get insights into the improvements enabled by each of the trainable components, we also evaluated a conventional structure where only the channel estimator was enhanced. The results indicate that the combined use of both the DL channel estimator and demapper is key to achieve a substantial reduction of the coded BERs across all scenarios.

We believe that such architectures, enabling performance improvements while remaining standard-compliant and reasonably complex, could be deployed in BSs for the next generation of wireless communication systems. However, it has be shown that additional gains can be still be achieved with a fully NN-based receiver that jointly processes all users [82], in contrast to our approach in which they are processed independently. Unlocking these gains while preserving the flexibility and interpretability of conventional architectures remains a significant challenge that is yet to be solved.

— 6 —

Conclusion and Future Directions

Summary of contributions

Over the course of this manuscript, we studied three strategies that aim at bringing the benefits of DL to the physical layer of communication systems. The first strategy, that we referred to as the NN-based block optimization, was used to design an NN that could replace current MU-MIMO detectors. Available works often exploit the deep unfolding approach, in which a traditional iterative receiver algorithm is unfolded and trainable parameters are inserted. Among these works, the MMNet detector showed impressive gains, but had to be retrained for each new channel realization. We therefore proposed in Chapter 3 to use a second NN, the hypernetwork, that takes as input a channel realization and generates a set of optimized parameters for an MMNet-based detector. To reduce the complexity of the system and the number of parameters that must be estimated, we both leveraged the QR-decomposition of the channel matrix and adopted a relaxed form of weight sharing. The hypernetwork training was carried out by backpropagating the gradients through the MMNet detector up to the hypernetwork trainable parameters. Simulation result demonstrated that the resulting architecture, referred to as HyperMIMO, achieves near state-of-the-art performance as long as the channel statistics does not change significantly. Other works subsequently reduced or eliminated some HyperMIMO downsides, for example by enabling scalability to any number of users and performance improvements on a wider range of channels. But these detectors are still trained to optimize their own performance only, which does not guaranty any system-level optimality.

The second strategy, in which the system is optimized end-to-end by implementing the transceivers as NNs, was discussed in Chapter 4. It allows for a deeper optimization of the

transmit and receive processing since it does not have to comply with existing protocols. We applied this approach to the design of OFDM waveforms tailored for specific needs. Specifically, we derived an NN architecture and a corresponding training algorithm that maximize an achievable rate while satisfying PAPR and ACLR constraints. This was achieved by expressing the objective and the constraints as differentiable functions that can be minimized using SGD. The constrained optimization problem was then solved using the augmented Lagrangian method. Numerical results demonstrated the effectiveness of the proposed approach, with trained systems that meet ACLR and PAPR targets, does not require pilot signals, and achieve throughputs comparatively higher than a TR baseline. These gains are driven by the emergence of new high-dimensional and asymmetrical constellations, in which the position and the energy associated with each symbol is a function of the subcarrier index and of all other symbols transmitted simultaneously. Although promising, the corresponding transmitter and receiver are too complex for any short-term implementation, does not satisfy current standards, and are limited to SISO transmissions. These limits pave the way for future research.

Finally, a third hybrid strategy was proposed in Chapter 5, and consists in inserting DL components into a traditional architecture that is trained in an end-to-end manner. We leveraged this strategy to enhance a conventional MO-MIMO receiver with three CNNs. The first and second ones learn the channel estimation error statistics from the data, which improve the symbol detection accuracy. The third one performs the bit demapping on the entire RG so that it can estimate and correct the distortions present on the equalized symbols. The end-to-end training also allows the system to be optimized on an achievable transmission rate. Simulations were performed with 3GPP-compliant channels with different speed ranges and two pilot patterns. The results indicate that our receiver achieves gains across all scenarios, both in uplink and downlink, and especially at high speeds. More importantly, it preserves the scalability of conventional architectures, and is composed of components that are individually interpretable and of reasonable complexity. Such systems, trained to handle multiple modulations and pilot patterns, could become standard-compliant as the DL components are only inserted in the receiver. Finally, and in contrast to many comparable works, perfect channel estimates are not needed during training. Overall, this strategy seems more suitable for practical use in the near future.

Future directions

The deployment of DL in the physical layer will probably be conducted in several phases. The first one corresponds to the integration of DL-based blocks into traditional BS receivers, trained following either the block-based or the hybrid strategy. The success of this integration is first tied to the availability of dedicated DL accelerators that can run at sufficient speeds. Fortunately, such processing units are already being designed to be implemented on BSs, where the constraints on the chip size and energy consumption are less stringent than on mobile devices. But DL is first and foremost about data, so that the gains provided by its integration are also tied to our capacity to build sufficient datasets. Indeed, most published works focus on

off-line training using channel simulators, which might not be representative of every scenario that will be encountered in real life. Collecting representative datasets from many diverse environments is therefore essential to ensure the reliability of these DL components.

These challenges are exacerbated on mobile devices, in which the integration of DL might correspond to a second deployment phase. To limit the amount of on-device computations, DL components will probably be trained in a centralized manner. This both requires that data be regularly collected from the devices and that updated parameters be transferred to them. On the one hand, such periodic data transfers will generate energy and bandwidth consumption, in addition to raising confidentiality and privacy concerns. On the other hand, they will enable more flexible communications, as the transceivers could be adapted to any specific hardware and environment. This should lead to improved gains since the DL models will be able to embrace distortions and channel specificities that are not present in traditional models. Such flexibility and adaptability are especially interesting for upcoming 6G networks, which are expected to support a wide range of use-case such as vehicular communications or small-scale sub-networks.

The increasing availability of DL accelerators at both ends of the transmission should eventually allow the emergence of fully NN-based transceivers, trained using the end-to-end strategy. Many predict that this third phase will simplify the set of different options and parameters used in current standards, as each transmitter-receiver pair could be continuously optimized to maximize its performance. However, the benefits provided by NN-based systems are jointly tied to the chosen NN architectures, training procedures, and to the quality of the available data. In industry sectors in which AI already plays an important role, such as voice assistants or autonomous driving, companies are usually reluctant to give any details about their specific NNs since they are at the core of their technical advances. The joint training of transmitters and receivers designed by different manufacturers therefore represents another major challenge for the telecommunication industry. Future standards need to shift from regulating the behavior of communication algorithms to allowing the end-to-end optimization of NN-based systems. Shared datasets and procedures will also be needed to enable proper testing, as the black-box nature of NNs prevents the derivation of the performance guarantees traditionally offered by model-based systems.

In its Ph.D.dissertation published in 2000, Joseph Mitola III described the concept of cognitive radio as "*the point in which wireless personal digital assistants (PDAs) and the related networks are sufficiently computationally intelligent about radio resources and related computer-to-computer communications to detect user communications needs as a function of use context, and to provide radio resources and wireless services most appropriate to those needs*" [96]. The next phases of DL deployment in communication systems could therefore be focused on the joint training of NNs for the physical and medium access control (MAC) layers, finally giving rise to the first fully AI-based cognitive radios.

— A —

Grouped-LMMSE Equalizer

We aim to find the LMMSE matrix to equalize a group of REs that spans multiple symbols $m \in [M_b, M_e]$ and subcarriers $n \in [N_b, N_e]$, with $1 \leq M_b \leq M_e \leq M$ and $1 \leq N_b \leq N_e \leq N$. Let us denote by $\widehat{\mathbf{H}}_{m,n}$ the channel estimated at a RE (m, n) and by $\widetilde{\mathbf{H}}_{m,n}$ the corresponding estimation errors. It is assumed that $\widehat{\mathbf{H}}$ is known, but that $\widetilde{\mathbf{H}}$, the symbols, and the noise, conditioned on the channel estimates, are random and uncorrelated. The channel transfer function for that group is

$$\mathbf{y}_{m,n} = \left(\widehat{\mathbf{H}}_{m,n} + \widetilde{\mathbf{H}}_{m,n}\right) \mathbf{x}_{m,n} + \mathbf{n}_{m,n}, \quad \forall m \in [M_b, M_e], n \in [N_b, N_e]. \tag{A.1}$$

We denote by $\mathbf{W}_{m,n},\ m \in [M_b, M_e], n \in [N_b, N_e]$, the LMMSE matrix that minimizes

$$\begin{aligned} \mathcal{L}(\mathbf{W}_{m,n}) &= \sum_{m'=M_b}^{M_e} \sum_{n'=N_b}^{N_e} \mathrm{MSE}\left(\mathbf{x}_{m',n'}, \mathbf{W}_{m,n}\mathbf{y}_{m',n'}\right) \\ &= \mathbb{E}\left[\sum_{m'=M_b}^{M_e} \sum_{n'=N_b}^{N_e} \left(\mathbf{x}_{m',n'} - \mathbf{W}_{m,n}\mathbf{y}_{m',n'}\right)\left(\mathbf{x}_{m',n'} - \mathbf{W}_{m,n}\mathbf{y}_{m',n'}\right)^{\mathsf{H}}\right]. \end{aligned} \tag{A.2}$$

Therefore, $\mathbf{W}_{m,n}$ nulls the gradient

$$\nabla_{\mathbf{W}_{m,n}} \mathcal{L}(\mathbf{W}_{m,n}) = 2\mathbf{W}\mathbb{E}\left[\sum_{m'=M_b}^{M_e} \sum_{n'=N_b}^{N_e} \mathbf{y}_{m',n'}\mathbf{y}_{m',n'}^{\mathsf{H}}\right] - 2\mathbb{E}\left[\sum_{m'=M_b}^{M_e} \sum_{m'=N_b}^{N_e} \mathbf{x}_{m',n'}\mathbf{y}_{m',n'}^{\mathsf{H}}\right] \stackrel{!}{=} 0 \tag{A.3}$$

which leads to

$$\begin{aligned}\mathbf{W}_{m,n} &= \left(\mathbb{E}\left[\sum_{m'=M_b}^{M_e}\sum_{n'=N_b}^{N_e}\mathbf{x}_{m',n'}\mathbf{y}_{m',n'}^{\mathsf{H}}\right]\right)\left(\mathbb{E}\left[\sum_{m'=M_b}^{M_e}\sum_{n'=N_b}^{N_e}\mathbf{y}_{m',n'}\mathbf{y}_{m',n'}^{\mathsf{H}}\right]\right)^{-1} \qquad \text{(A.4)}\\ &= \left(\sum_{m'=M_b}^{M_e}\sum_{n'=N_b}^{N_e}\widehat{\mathbf{H}}_{m',n'}^{\mathsf{H}}\right)\left(\sum_{m'=M_b}^{M_e}\sum_{n'=N_b}^{N_e}\left(\widehat{\mathbf{H}}_{m',n'}\widehat{\mathbf{H}}_{m',n'}^{\mathsf{H}}+\mathbb{E}\left[\widetilde{\mathbf{H}}_{m',n'}\widetilde{\mathbf{H}}_{m',n'}^{\mathsf{H}}\right]+\sigma^2\mathbf{I}_{N_m}\right)\right)^{-1}.\end{aligned}$$

— B —

Résumé en Français

B.1 Introduction

B.1.1 Quand l'Apprentissage Profond Rencontre le Traitement du Signal

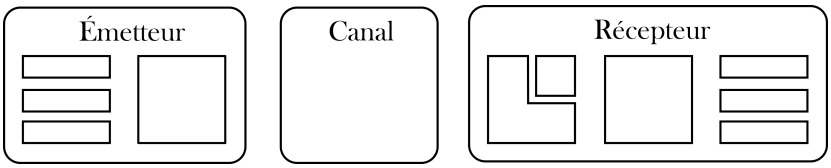
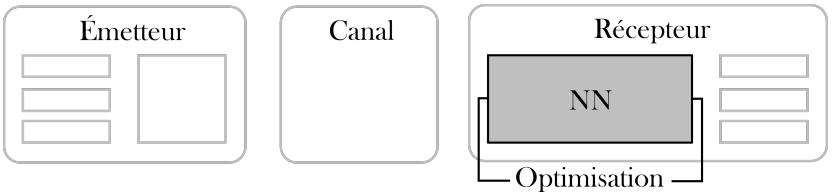
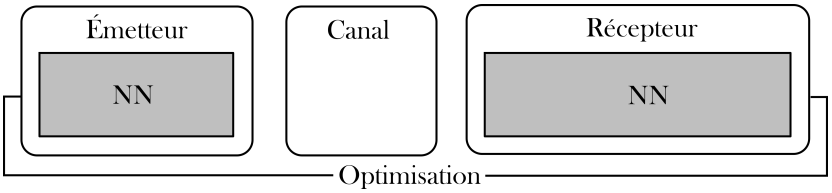


FIGURE B.1 : Un système de communication traditionnel basé sur des blocs.

Le premier modèle de réseau de neurones (neural network, NN) a été inventé en 1943 [1], mais soixante années de recherche et une importante augmentation de la puissance de calcul ont été nécessaires pour permettre une adoption massive de l'apprentissage profond (deep learning, DL) par l'industrie [2]. Dans le même temps, de nouvelles générations de systèmes de communication sont apparues tous les dix ans, à partir de 1979 [9]. Au fur et à mesure que les émetteurs et les récepteurs devenaient de plus en plus complexes, la tractabilité a été obtenue en divisant les chaînes de traitement d'émission et de réception en petits composants, généralement appelés blocs de traitement et illustrés sur la Figure B.1. Ces systèmes de communication basés sur des blocs souffrent de multiples inconvénients. D'un côté, les modèles de canal simplistes ne parviennent pas à capturer toutes les spécificités du matériel et des phénomènes de propagation sous-jacents. D'un autre côté, l'optimisation conjointe de l'émetteur et du récepteur devient rapidement intractable lorsque des modèles de canaux plus réalistes sont utilisés, et donc l'optimisation de chaque bloc est généralement réalisée indépendamment. Cela ne garantit pas l'optimalité du système résultant, comme démontré pour les blocs de codage de canal et de modulation [10]. Enfin, de la signalisation est souvent nécessaire entre l'émetteur et le récepteur, ce qui introduit une surcharge qui réduit le débit du système.



(a) Optimisation des blocs basée sur un NN : un NN est optimisé pour remplacer un ou plusieurs blocs dans un système de communication.



(b) Optimisation de bout en bout : des émetteurs-récepteurs basés sur les NN sont optimisés pour maximiser les performances de bout en bout d'un système.

FIGURE B.2 : Différents niveaux d'intégration des NNs dans les systèmes de communication.

Deux stratégies principales se dessinent concernant le DL dans la couche physique. La première approche correspond à une stratégie d'optimisation des blocs basée sur un NN, comme le montre la Figure B.2a, dans laquelle un ou plusieurs blocs de traitement consécutifs sont remplacés par un NN qui est entraîné indépendamment des autres blocs. La deuxième approche consiste à implémenter l'émetteur et le récepteur sous forme de NNs qui sont entraînés conjointement afin de maximiser les performances du système de bout en bout [13], [14]. Cette approche est souvent appelée stratégie d'optimisation de bout en bout, comme le montre la Figure B.2b. Cette stratégie permet aux systèmes d'être entièrement optimisés à partir de données récoltées sur le terrain, ce qui permet de traiter efficacement les dégradations matérielles et autres distorsions du canal sans nécessiter de modèle mathématique [15]. Cependant, chaque stratégie a ses défauts : les NNs basés sur des blocs (Figure B.2a) ne sont pas entraînés pour maximiser la performance globale du système. De plus les émetteurs-récepteurs entièrement appris (Figure B.2b) manquent d'interprétabilité et d'adaptabilité à un nombre variable d'utilisateurs. Cette thèse vise donc à apporter une réponse à la question de l'intégration optimale des composants DL dans les systèmes de communication sans fil.

B.1.2 Défis Actuels et Contributions de ce Travail

La prochaine génération de réseaux cellulaires devra prendre en charge un nombre croissant de services et de dispositifs différents [25]. À cette fin, les systèmes multi-utilisateurs à entrées et sorties multiples (multi-user multiple-input multiple-output, MU-MIMO) permettent un partage plus efficace des ressources disponibles. Les deux principaux défis liés au déploiement de ces systèmes sont la complexité de l'algorithme de détection de symboles et la dégradation des performances sur des canaux mal conditionnés. Ces dernières années, plusieurs approches ont tenté de relever ces défis en implémentant le détecteur comme un NN, ce qui correspond à la

stratégie d'optimisation par blocs basée sur des NNs. Cependant, elles obtiennent toujours des performances insatisfaisantes sur les canaux spatialement corrélés, ou sont exigeantes en termes de calcul puisqu'elles nécessitent un réentraînement pour chaque réalisation de canal. Dans ce travail, nous abordons ces deux problèmes en utilisant un réseau de neurones supplémentaire, appelé hyper-réseau, qui prend en entrée la matrice de canal et génère des poids optimisés pour un détecteur basé sur un NN. Une autre direction de recherche est l'amélioration de la forme d'onde liée au multiplexage par répartition en fréquences orthogonales (orthogonal frequency-division multiplexing, OFDM), qui souffre de multiples inconvénients, tels qu'un rapport élevé de puissance de crête à puissance moyenne (peak-to-averge power ratio, PAPR) et un rapport de fuite dans les cannaux adjacents (adjacent channel leakage ratio, ACLR). Pour résoudre ces problèmes, nous tirons parti de la stratégie d'optimisation de bout en bout et modélisons l'émetteur et le récepteur comme deux NNs qui implémentent un schéma de modulation à haute dimension et estiment les bits transmis.

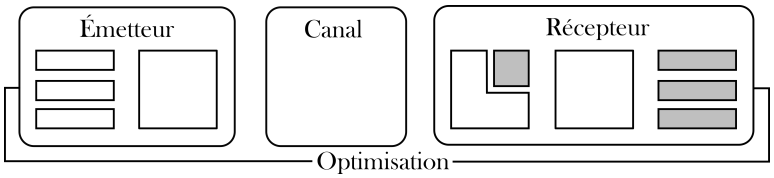


FIGURE B.3 : Une stratégie d'optimisation hybride.

Cependant, ces systèmes de bout en bout manquent d'interprétabilité et d'adaptabilité, ce qui est particulièrement important dans les transmissions MU-MIMO puisque l'algorithme de réception doit permettre une adaptation facile à un nombre variable d'utilisateurs. Pour cette raison, nous proposons un récepteur MU-MIMO amélioré par DL qui s'appuie sur une architecture conventionnelle pour préserver son interprétabilité et son adaptabilité. Cette approche peut être considérée comme une stratégie hybride, dans laquelle plusieurs composants basés sur du DL sont insérés dans une architecture traditionnelle basée sur des blocs, mais sont optimisés pour maximiser les performances du système de bout en bout (Figure B.3).

Notations

Les tenseurs et les matrices sont désignés par des lettres majuscules en gras et les vecteurs par des lettres minuscules en gras. Nous désignons respectivement par $\mathbf{m}_a$ et $m_{a,b}$ le vecteur et le scalaire formés par le découpage de la matrice $\mathbf{M}$ le long de sa première et de ses deux premières dimensions. Similairement, nous désignons par $\mathbf{T}_{a,b} \in \mathbb{C}^{N_c \times N_d}$ ($\mathbf{t}_{a,b,c} \in \mathbb{C}^{N_d}$, $t_{a,b,c,d} \in \mathbb{C}$) la matrice (le vecteur, le scalaire) formé par le découpage du tenseur $\mathbf{T} \in \mathbb{C}^{N_a \times N_b \times N_c \times N_d}$ le long de ses deux (trois, quatre) premières dimensions. La notation $\mathbf{T}^{(k)}$ indique que la quantité en question n'est considérée que pour le $k^{\text{ième}}$ utilisateur. $\odot$, $(\cdot)^{\mathsf{T}}$ et $(\cdot)^{\mathsf{H}}$ désignent respectivement le produit par éléments, la transposition, et la transposition conjuguée.

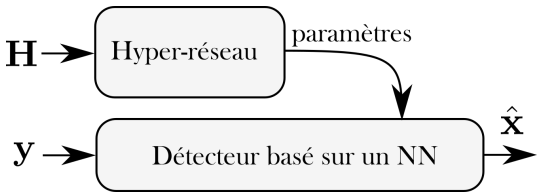


FIGURE B.4 : HyperMIMO : un hyper-réseau génère les poids d'un détecteur basé sur un NN.

B.2 HyperMIMO : un Détecteur MIMO Basé sur un Hyper-réseau

B.2.1 Motivations

Comme nous l'avons vu précédemment, la détection optimale dans les systèmes MIMO est connue pour être NP-difficile [37], et les approches moins complexes souffrent généralement de performances insatisfaisantes sur des canaux corrélés. Récemment, des progrès dans la détection MIMO ont été réalisés en utilisant le DL pour améliorer le bloc d'égalisation, ce qui correspond à une optimisation basée sur un bloc. Une possibilité est d'ajouter des paramètres entraînables à ces algorithmes itératifs et d'interpréter l'ensemble de la structure comme un NN [45], [46], mais la plupart des approches souffrent toujours d'une baisse de performance sur des canaux corrélés. Ce problème a été atténué par le détecteur MMNet [47], qui atteint de bonnes performances sur ces canaux. Cependant, le besoin de réapprentissage pour chaque réalisation de canal rend son utilisation difficile.

Dans ce qui suit, nous atténuons ce problème en exploitant l'idée émergente de l'hyper-réseaux [48], [49]. Appliqué à notre configuration, elle consiste à avoir un NN secondaire, appelé hyper-réseau, qui génère pour une matrice de canal donnée un ensemble optimisé de poids pour un détecteur basé sur un NN. Ce schéma, que nous avons appelé HyperMIMO dans notre article d'introduction [50], est illustré dans la Figure B.4. Nous avons évalué l'approche proposée en utilisant des simulations sur des canaux spatialement corrélés. Nos résultats montrent qu'HyperMIMO atteint une performance proche de celle d'MMNet entraîné pour chaque réalisation de canal, et surpasse l'OAMPNet récemment proposé [46].

B.2.2 Cadre de Travail

Nous considérons un canal de liaison montante MU-MIMO classique, où K utilisateurs à antenne unique communiquent avec une station de base (base station, BS) ayant L antennes de réception. La fonction de transfert du canal est

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n} \tag{B.1}$$

où $\mathbf{x} \in \mathcal{C}^K$ est le vecteur des symboles transmis, $\mathbf{y} \in \mathbb{C}^L$ est le vecteur des symboles déformés reçus, $\mathbf{H} \in \mathbb{C}^{L\times K}$ est la matrice de canal, et $\mathbf{n} \sim \mathcal{CN}(\mathbf{0}, \sigma^2\mathbf{I}_L)$ est le bruit gaussien complexe indépendant et identiquement distribué (i.i.d.) avec une puissance σ^2 dans chaque dimension

complexe. On suppose que $\mathbf{H}$ et σ sont parfaitement connus du récepteur. Dans la suite, nous considérons le problème de la détection dure de symboles, dans lequel le symbole estimé $\hat{\hat{\mathbf{x}}}$ doit appartenir à la constellation utilisée, c'est-à-dire, $\hat{\hat{\mathbf{x}}} \in \mathcal{C}^K$.

De multiples schémas DL ont été proposés pour obtenir un meilleur compromis performance-complexité que les détecteurs à maximum de vraisemblance ou à erreur quadratique moyenne minimale linéaire (linear minimum mean square error, LMMSE). Une technique prometteuse, appelée *deep unfolding*, consiste à améliorer les schémas itératifs existants en ajoutant des paramètres entraînables, et en entraînant le tout comme un NN. Généralement, chaque itération comprend une étape linéaire suivie d'une étape de débruitage non linéaire :

$$\begin{aligned} \boldsymbol{\kappa}^{(i)} &= \hat{\mathbf{x}}^{(i)} + \mathbf{A}^{(i)} \left(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}^{(i)} + \mathbf{c}^{(i)} \right) \\ \hat{\mathbf{x}}^{(i+1)} &= \chi^{(i)} \left(\boldsymbol{\kappa}^{(i)}, \tau^{(i)} \right) \end{aligned} \tag{B.2}$$

où l'exposant (i) est utilisé pour faire référence à la $i^{\text{ème}}$ itération et $\hat{\mathbf{x}}^{(0)}$ est fixé à $\mathbf{0}$. $\tau^{(i)}$ désigne la variance estimée des composantes du vecteur bruit $\boldsymbol{\kappa}^{(i)} - \mathbf{x}^{(i)}$ à l'entrée du débruiteur, qui est supposé être i.i.d.. Les algorithmes itératifs diffèrent par leurs choix de matrices $\mathbf{A}^{(i)} \in \mathbb{C}^{K \times L}$, de vecteurs de biais $\mathbf{c}^{(i)} \in \mathbb{C}^K$, et de fonctions de débruitage $\chi^{(i)}(\cdot)$. Une limitation de la plupart des schémas de détection est leur faible performance sur les canaux corrélés. OAMPNet [46] apporte des améliorations en ajoutant deux paramètres entraînables par itération. MMNet [47] va plus loin en rendant toutes les matrices $\mathbf{A}^{(i)}$ entraînables et en relâchant la contrainte selon laquelles les $\boldsymbol{\kappa}^{(i)} - \mathbf{x}^{(i)}$ doivent être identiquement distribué. Bien qu'MMNet atteigne des performances de pointe sur les canaux spatialement corrélés, il doit être réentraîné pour chaque matrice de canal, ce qui le rend peu pratique. L'idée clé de notre approche est donc de remplacer le processus d'entraînement requis par MMNet pour chaque réalisation de canal par une seule inférence à travers un hyper-réseau entraîné.

B.2.3 HyperMIMO

Pour réduire le nombre de paramètres d'MMNet, nous réalisons la décomposition QR de la matrice de canal, $\mathbf{H} = \mathbf{QR}$, où $\mathbf{Q}$ est une matrice orthogonale de dimension $L \times L$ et $\mathbf{R}$ est une matrice triangulaire supérieure de dimension $L \times K$. Nous supposons que $L > K$, et par conséquent $\mathbf{R} = \begin{bmatrix} \mathbf{R_A} \\ \mathbf{0} \end{bmatrix}$ où $\mathbf{R_A}$ a pour dimension $K \times K$, et $\mathbf{Q} = [\mathbf{Q_A} \boldsymbol{Q_B}]$ où $\mathbf{Q_A}$ a pour dimension $L \times K$. Nous définissons $\bar{\mathbf{y}} \coloneqq {\mathbf{Q_A}}^H \mathbf{y}$ et $\bar{\mathbf{n}} \coloneqq {\mathbf{Q_A}}^H \mathbf{n}$, et réécrivons (B.1) comme suit

$$\bar{\mathbf{y}} = \mathbf{R_A}\mathbf{x} + \bar{\mathbf{n}}. \tag{B.3}$$

Notez que $\bar{\mathbf{n}} \sim \mathcal{CN}(\mathbf{0}, \sigma^2 \mathbf{I}_K)$. MMNet définit $\mathbf{c}^{(i)}$ à $\mathbf{0}$ pour tous les i et utilise le même débruiteur pour toutes les itérations, qui sont définies par

$$\begin{aligned} \boldsymbol{\kappa}^{(i)} &= \hat{\mathbf{x}}^{(i)} + \boldsymbol{\Theta}^{(i)} \left(\bar{\mathbf{y}} - \mathbf{R_A}\hat{\mathbf{x}}^{(i)} \right) \\ \hat{\mathbf{x}}^{(i+1)} &= \chi \left(\boldsymbol{\kappa}^{(i)}, \boldsymbol{\tau}^{(i)} \right) \end{aligned} \tag{B.4}$$

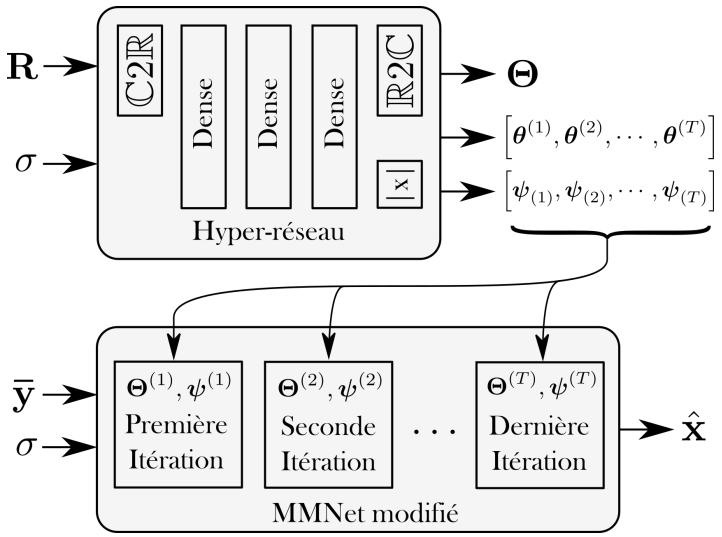


FIGURE B.5 : Architecture détaillée d'HyperMIMO.

où $\mathbf{\Theta}^{(i)}$ est une matrice complexe de dimension $K \times K$ dont les éléments doivent être optimisées pour chaque réalisation de canal. Le principal avantage de l'utilisation de la décomposition QR est que la dimension des matrices $\mathbf{\Theta}^{(i)}$ à optimiser est de $K \times K$ au lieu de $K \times L$, qui est la dimension $\mathbf{A}^{(i)}$ dans (B.2). Ceci est significatif puisque le nombre d'utilisateurs actifs K est généralement beaucoup plus petit que le nombre d'antennes L de la BS. MMNet se compose de I couches exécutant (B.4), et une décision dure est prise pour prédire l'estimation finale $\hat{\hat{\mathbf{x}}}$.

La Figure B.5 montre en détail l'architecture d'HyperMIMO. Comme notre variante d'MMNet opère sur $\bar{\mathbf{y}}$, l'hyper-réseau est alimenté par $\mathbf{R_A}$ et l'écart type du bruit du canal σ. Notez que, comme $\mathbf{R_A}$ est triangulaire supérieur, seuls $K(K+1)/2$ éléments non nuls doivent être fournis à l'hyper-réseau. Pour diminuer encore le nombre de sorties de l'hyper-réseau, nous adoptons une forme relaxée de partage de poids inspirée de [48]. Au lieu de calculer les éléments de chaque $\mathbf{\Theta}^{(i)}, i = 1, \ldots, I$, l'hyper-réseau estime une seule matrice $\mathbf{\Theta}$ ainsi que I vecteurs $\boldsymbol{\theta}^{(i)} \in \mathbb{R}^K$. Pour chaque itération i, $\mathbf{\Theta}^{(i)}$ est calculé par

$$\mathbf{\Theta}^{(i)} = \mathbf{\Theta}\left(\mathbf{I}_K + \operatorname{diag}\left(\boldsymbol{\theta}^{(i)}\right)\right). \tag{B.5}$$

Comme $\mathbf{R_A}$ est à valeur complexe, une couche $\mathbb{C}2\mathbb{R}$ transforme les éléments complexes de $\mathbf{R_A}$ en éléments réels, en concaténant les parties réelles et imaginaires des éléments complexes. Pour générer une matrice $\mathbf{\Theta}$ à valeur complexe, une couche $\mathbb{R}2\mathbb{C}$ effectue l'opération inverse de $\mathbb{C}2\mathbb{R}$. L'hyper-réseau doit également calculer les valeurs des I vecteurs $\boldsymbol{\psi}^{(i)}$. Comme les éléments de ces vecteurs doivent être positifs, une fonction d'activation à valeur absolue est utilisée dans la dernière couche, comme le montre la Figure B.5. HyperMIMO, qui comprend l'hyper-réseau et MMNet, est entraîné en minimisant l'erreur quadratique moyenne (mean squared error, MSE) entre les symboles transmis et estimés. Lors de l'apprentissage d'HyperMIMO, l'hyper-réseau et MMNet forment un seul NN, de sorte que la sortie de l'hyper-réseau constitue les poids du détecteur MMNet. Les seuls paramètres entraînables sont donc ceux de l'hyper-réseau.

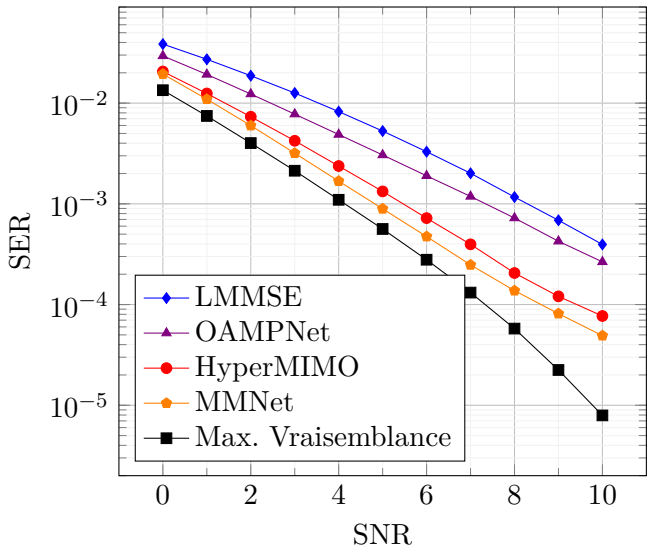
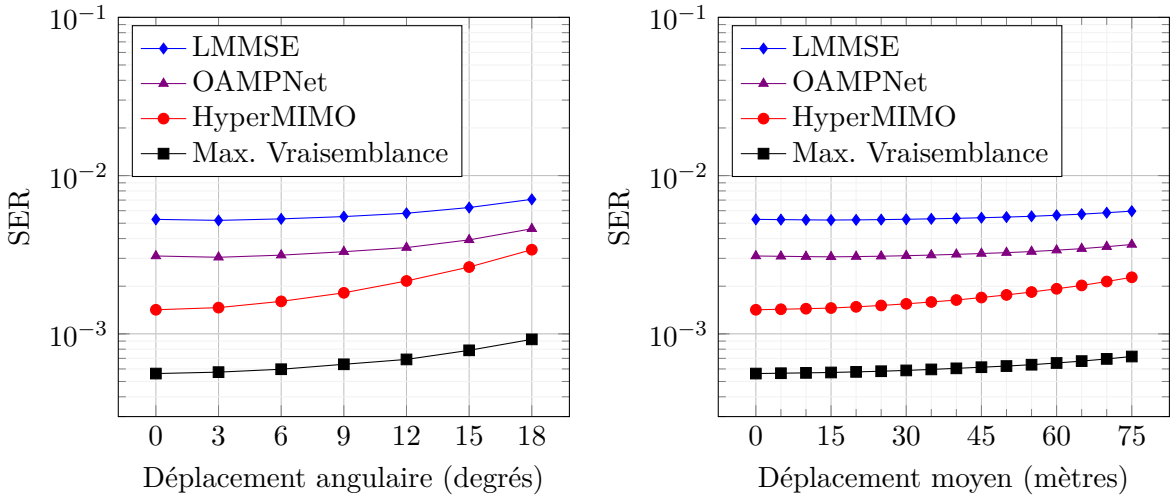


FIGURE B.6 : SER atteint par différents systèmes.



(a) Mobilité angulaire.

(b) Mobilité aléatoire 2D.

FIGURE B.7 : SER obtenu par les approches comparées en cas de mobilité.

B.2.4 Résultats des Simulations

On considère le modèle de diffusion locale avec corrélation spatiale présenté dans [26, Ch. 2.6]. On suppose une répartition parfaite de la puissance, ce qui fait que tous les utilisateurs sont à la même distance r de la BS et que le gain moyen est de un. Le rapport signal/bruit (signal-to-noise ration, SNR) de la transmission est défini par la SNR $= \frac{\mathbb{E}\left[\frac{1}{N_r}\|\mathbf{y}\|_2^2\right]}{\sigma^2} = \frac{1}{\sigma^2}$. Le nombre d'antennes qui équipent la BS a été fixé à $L = 12$, et le nombre d'utilisateurs à $K = 6$.

Pour une certaine répartition des utilisateurs, HyperMIMO a été entraîné en échantillonnant de manière aléatoire les matrices de canal $\mathbf{H}$, les SNRs dans la plage [0,10] dB, et les symboles d'une constellation de modulation QPSK pour chaque utilisateur.

La Figure B.6 montre le taux d'erreur de symboles (symbol error rate, SER) obtenu par HyperMIMO, LMMSE, OAMPNet avec 10 itérations, MMNet avec 10 itérations et entraîné pour chaque réalisation de canal, et le détecteur de maximum de vraisemblance. Comme prévu, MMNet, lorsqu'il est entraîné pour chaque réalisation de canal, atteint une performance proche de celle du maximum de vraisemblance. HyperMIMO atteint un SER légèrement supérieur à MMNet, mais inférieur à OAMPNet et LMMSE. La robustesse d'HyperMIMO à la mobilité des utilisateurs a été testée en évaluant le SER obtenu lorsque les utilisateurs subissent une mobilité angulaire (Figure B.7a) ou se déplacent dans des directions 2D aléatoires (Figure B.7b) à partir des positions pour lesquelles le système a été entraîné. On peut voir que le SER obtenu par HyperMIMO se dégrade gracieusement lorsque le déplacement angulaire augmente, et ne devient jamais pire que LMMSE ou OAMPNet. De même, le SER obtenu par HyperMIMO se dégrade gracieusement lorsque la distance de déplacement augmente. Ces résultats montrent que, bien qu'ayant été entraîné pour un ensemble particulier de positions d'utilisateurs, HyperMIMO reste relativement robuste à la mobilité.

B.2.5 Conclusion

Dans cette section, nous avons proposé de tirer parti de l'idée des hyper-réseaux pour éviter de devoir réentraîner un détecteur MMNet pour chaque réalisation de canal tout en obtenant des performances compétitives. Pour réduire la complexité de l'hyper-réseau, MMNet a été modifié pour diminuer le nombre de paramètres entraînables, et une forme de partage des poids a été utilisée. Les simulations ont révélé que l'architecture HyperMIMO résultante atteint des performances proches de l'état de l'art dans des canaux hautement corrélés lorsqu'elle est entraînée et évaluée avec le même nombre d'utilisateurs et avec des statistiques de canal fixes. Cependant, bien que ces détecteurs basés sur des NNs représentent des améliorations prometteuses par rapport aux algorithmes traditionnels, leur optimisation basée sur les blocs ne garantit toujours pas l'optimalité globale du récepteur. Un autre inconvénient est que la plupart des détecteurs basés sur des NNs nécessitent des estimations parfaites du canal lors de l'entraînement, mais elles ne sont généralement pas disponibles dans la pratique. Pour ces raisons, nous nous concentrons dans le chapitre suivant sur l'optimisation de bout en bout du couple émetteur-récepteur, mais uniquement pour les systèmes à entrée unique et sortie unique (single-input single-output, SISO).

B.3 Apprentissage de Formes d'Onde OFDM avec des Contraintes de PAPR et d'ACLR

B.3.1 Motivation

Le multiplexage spatial ne peut pas être exploité dans les systèmes SISO, et d'autres approches doivent donc être envisagées. L'une d'entre elles consiste à concevoir de nouvelles formes d'onde qui répondent à des exigences plus strictes concernant les caractéristiques du signal. Parmi les solutions possibles, l'OFDM est déjà utilisé dans la plupart des systèmes de communication modernes, mais il souffre d'un PAPR et d'un ACLR élevés, ce qui pourrait entraver son utilisation dans les systèmes post-5G. Comme les futures BS et les équipements utilisateurs devraient être équipés d'accélérateurs DL dédiés, de nombreux travaux ont proposé de concevoir des émetteurs-récepteurs basés sur des NNs et destinés à différents canaux. Par exemple, l'apprentissage des géométries de constellation pour réaliser des communications sans pilote et sans préfixe cyclique (cyclic prefix, CP) sur des canaux OFDM a été effectué dans [16], et la conception de systèmes basés sur des NNs pour les canaux multiporteuse avec *fading* et les communications par fibre optique ont été étudiés respectivement dans [74] et [75].

Dans ce qui suit, nous utilisons la stratégie d'optimisation de bout en bout pour concevoir des formes d'onde OFDM, approche que nous avons publié dans [64]. Plus précisément, cette stratégie est basée, d'une part, sur des émetteurs-récepteurs basés sur des réseaux de neurones convolutionels (convolutional neural networks, CNNs) et, d'autre part, sur une procédure d'apprentissage qui permet à la fois de maximiser un taux d'information réalisable tout en satisfaisant des contraintes de PAPR et d'ACLR. Le système de bout en bout est comparé à une implémentation presque idéale d'un système à réservation de tonalité (tone reservation, TR), dans lequel un certain nombre de sous-porteuses sont utilisées pour générer des signaux de réduction de puissance de crête. Les deux systèmes sont évalués sur un modèle de canal conforme aux modèles 3GPP. Les résultats de l'évaluation montrent que le système basé sur l'apprentissage permet d'atteindre les objectifs de PAPR et d'ACLR et tout en permettant des gains de débit allant de 3% à 30% par rapport au système de référence avec des caractéristiques similaires. À notre connaissance, cette méthode est la première approche basée sur du DL qui permet à la fois de maximiser un taux d'information des transmissions OFDM et de fixer des objectifs de PAPR et d'ACLR.

B.3.2 Description du Problème

Un canal OFDM est considéré, avec N sous-porteuses et un *slot* temporel, qui se compose de $M = 14$ symboles OFDM adjacents (Figure B.8). Dans cette section, les sous-porteuses sont indexées par l'ensemble $\mathcal{N} = \left\{-\frac{N-1}{2}, \cdots, \frac{N-1}{2}\right\}$, N étant supposé impair par commodité. Si l'on considère la grille de ressources (resource grid, RG) en entier, le canal OFDM peut être exprimé comme suit

$$\mathbf{Y} = \mathbf{H} \odot \mathbf{X} + \mathbf{N} \tag{B.6}$$

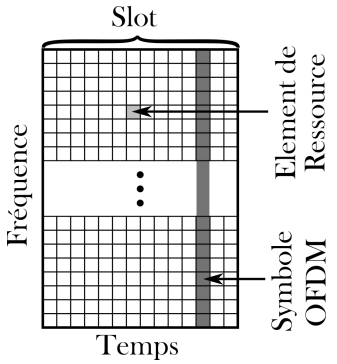


FIGURE B.8 : Une grille de ressources OFDM.

où $\mathbf{X} \in \mathbb{C}^{M\times N}$ et $\mathbf{Y} \in \mathbb{C}^{M\times N}$ représentent respectivement la matrice des symboles en bande de fréquence de base (frequency baseband symbol, FBS) envoyés et reçus, $\mathbf{H} \in \mathbb{C}^{M\times N}$ est la matrice des coefficients de canal, et $\mathbf{N} \in \mathbb{C}^{M\times N}$ est la matrice de bruit gaussien additif telle que chaque élément a une variance σ^2. Nous considérons un environnement qui varie lentement de sorte que le canal peut être supposé constant sur la durée d'un slot. La matrice des bits à transmettre sur le symbole OFDM m est notée $\mathbf{B}_m = [\mathbf{b}_{m,1}, \cdots, \mathbf{b}_{m,N}]^\mathsf{T}$, où $\mathbf{b}_{m,n} \in \{0,1\}^Q, 1 \leq m \leq M, 1 \leq n \leq N$, est un vecteur de bits à envoyer et Q est le nombre de bits par utilisation du canal. L'émetteur module chaque $\mathbf{B}_m$ sur les FBS $\mathbf{x}_m \in \mathbb{C}^N$, qui sont mappés sur les sous-porteuses orthogonales. Lorsque les CPs d'une durée T^{CP} sont ajoutés aux symboles OFDM de durée T, le spectre du signal devient

$$S_m^{\mathrm{CP}}(f) = \sum_{n\in\mathcal{N}} x_{m,n} \frac{1}{\sqrt{\Delta_f^{\mathrm{CP}}}} \mathrm{sinc}\left(\frac{f - n\Delta_f}{\Delta_f^{\mathrm{CP}}}\right) \tag{B.7}$$

où Δ_f est l'espacement entre les sous-porteuses, et $\Delta_f^{\mathrm{CP}} = \frac{1}{T+T^{\mathrm{CP}}}$. Le signal correspondant est

$$s(t) = \sum_{m=0}^{M-1} s_m(t) = \sum_{m=0}^{M-1} \sum_{n_i n\mathcal{N}} x_{m,n}\phi_n(t - mT^{\mathrm{tot}}) \tag{B.8}$$

où $T^{\mathrm{tot}} = T + T^{\mathrm{CP}}$ et les filtres de transmission $\phi_n(t), n \in \mathcal{N}$ sont définis comme :

$$\phi_n(t) = \frac{1}{\sqrt{T^{\mathrm{tot}}}} \mathrm{rect}\left(\frac{t}{T^{\mathrm{tot}}} - \frac{1}{2}\right) e^{j2\pi n \frac{t-T^{\mathrm{CP}}}{T}}. \tag{B.9}$$

Les formes d'onde OFDM présentent, entre autres, deux inconvénients majeurs. Le premier est leurs pics de forte amplitude, qui créent des distorsions dans le signal de sortie en raison de la saturation des amplificateurs de puissance. Désignons par $\nu(t) = \frac{|s(t)|^2}{\mathbb{E}[|s(t)|^2]}$ le rapport entre la puissance instantanée et la puissance moyenne d'un signal. Nous définissons le PAPR_ϵ comme le plus petit $e \geq 0$, tel que la probabilité que $\nu(t)$ soit plus grand que e soit inférieure à un

seuil $\epsilon \in (0, 1)$:

$$\mathrm{PAPR}_\epsilon \coloneqq \min e, \text{ s. t. } P\left(\nu(t) > e\right) \leq \epsilon. \tag{B.10}$$

Le réglage $\epsilon = 0$ conduit à la définition plus conventionnelle du PAPR $\frac{\max |s(t)|^2}{\mathbb{E}[|s(t)|^2]}$.

Le deuxième inconvénient de l'OFDM est son faible confinement spectral. Cette caractéristique est généralement mesurée par l'ACLR, qui est le rapport entre l'espérance de l'énergie hors bande $\mathbb{E}_{\mathbf{x}_m}[E_{O_m}]$ et l'eespérance de l'énergie dans la bande $\mathbb{E}_{\mathbf{x}_m}[E_{I_m}]$:

$$\mathrm{ACLR} \coloneqq \frac{\mathbb{E}_{\mathbf{x}_m}[E_{O_m}]}{\mathbb{E}_{\mathbf{x}_m}[E_{I_m}]} = \frac{\mathbb{E}_{\mathbf{x}_m}[E_{A_m}]}{\mathbb{E}_{\mathbf{x}_m}[E_{I_m}]} - 1 \tag{B.11}$$

où E_{O_m}, E_{I_m}, et $E_{A_m} = E_{O_m} + E_{I_m}$ sont respectivement l'énergie hors bande, dans la bande et l'énergie totale du symbole OFDM m. L'énergie dans la bande E_{I_m} est donnée par

$$E_{I_m} \coloneqq \int_{-\frac{N\Delta_f}{2}}^{\frac{N\Delta_f}{2}} |S_m(f)|^2 \, df = \mathbf{x}_m^H \mathbf{J} \mathbf{x}_m \tag{B.12}$$

où chaque élément $j_{a,b}$ de la matrice $\mathbf{J} \in \mathbb{R}^{N \times N}$ est

$$j_{a,b} = \frac{1}{\Delta_f^{\mathrm{CP}}} \int_{-\frac{N\Delta_f}{2}}^{\frac{N\Delta_f}{2}} \mathrm{sinc}\left(\frac{f - a\Delta_f}{\Delta_f^{\mathrm{CP}}}\right) \mathrm{sinc}\left(\frac{f - b\Delta_f}{\Delta_f^{\mathrm{CP}}}\right) df, \quad a, b \in \mathcal{N}. \tag{B.13}$$

Enfin, l'énergie totale peut être calculée plus facilement dans le domaine temporel :

$$E_{A_m} \coloneqq \int_{-\frac{T}{2}}^{\frac{T}{2}} |s(t)|^2 \, dt = \mathbf{x}_m^H \mathbf{K} \mathbf{x}_m \tag{B.14}$$

où $\mathbf{K} \in \mathbb{R}^{N \times N}$ a pour éléments

$$k_{a,b} = \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} e^{i2\pi(a-b)t/T} dt, \quad a, b \in \mathcal{N}. \tag{B.15}$$

Système de référence

L'une des techniques permettant de réduire le PAPR d'un signal OFDM est la TR, dans laquelle un sous-ensemble de tonalités (sous-porteuses) R est utilisé pour créer des signaux de réduction de crête. Ces sous-porteuses sont appelées tonalités de réduction de crête (peak reduction tones, PRTs), et les sous-porteuses restantes sont utilisées pour la transmission des données et des pilotes. Dans cette thèse, nous utilisons un solveur complexe pour trouver les signaux de réduction de crête optimaux pour chaque symbole OFDM. Plus de détails le systeme de référence utilisant la TR peuvent être trouvés dans la Section 4.2.2.

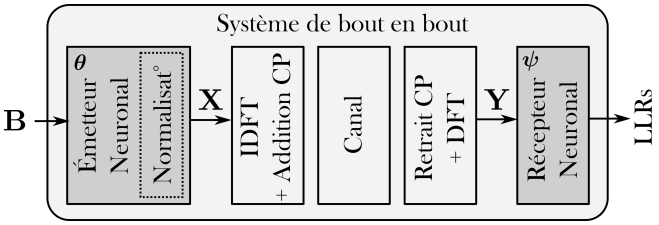


FIGURE B.9 : Système entraînable, où les blocs grisés représentent les composants entraînables.

B.3.3 Apprentissage d'une Modulation de Grande Dimension

Dans ce qui suit, nous formons un émetteur et un récepteur basés sur des NNs de bout en bout. Ce système est appelé système *end-to-end*, ou "E2E" par souci de concision, et est représenté schématiquement sur la Figure B.9. Nous cherchons à trouver une modulation de grande dimension et un détecteur associé qui maximisent le taux d'information pour la transmission et satisfont des contraintes sur le PAPR et l'ACLR du signal. L'émetteur et le récepteur du système fonctionnent au-dessus de l'OFDM, c'est-à-dire qu'une IDFT (DFT) est effectuée et qu'un préfixe cyclique est ajouté (supprimé) avant (après) la transmission (voir la Figure B.9). Nous considérons un taux d'information réalisable [16] qui dépend des paramètres entraînables de l'émetteur et du récepteur respectivement désignés par $\boldsymbol{\theta}$ et $\boldsymbol{\psi}$:

$$C(\boldsymbol{\theta}, \boldsymbol{\psi}) = \frac{1}{MN} \sum_{m \in \mathcal{M}} \sum_{n \in \mathcal{N}} \sum_{q=0}^{Q-1} I\left(b_{m,n,q}; \mathbf{y}_m | \, \boldsymbol{\theta}\right) \tag{B.16}$$
$$- \frac{1}{MN} \sum_{m \in \mathcal{M}} \sum_{n \in \mathcal{N}} \sum_{q=0}^{Q-1} \mathbb{E}_{\mathbf{y}_m} \left[\mathrm{D}_{\mathrm{KL}} \left(P(b_{m,n,q}|\mathbf{y}_m) || \widehat{P}_{\boldsymbol{\psi}}(b_{m,n,q}|\mathbf{y}_m) \right) \right]$$

où le premier terme est l'information mutuelle entre $b_{m,n,q}$ et $\mathbf{y}_m$, et le deuxième terme est la divergence K-L entre entre les vraies probabilités sur les bits et celles estimées par le démappeur basé sur un NN. Pour garantir une énergie moyenne unitaire par élément de ressource (resource element, RE), une couche de normalisation est ajoutée à l'émetteur (voir la Figure B.9). Nous pouvons désormais formuler le problème d'optimisation sous contrainte que nous voulons résoudre :

$$\underset{\boldsymbol{\theta}, \boldsymbol{\psi}}{\text{maximize}} \quad C(\boldsymbol{\theta}, \boldsymbol{\psi}) \tag{B.17a}$$
$$\text{subject to} \quad \mathrm{PAPR}_{\epsilon}(\boldsymbol{\theta}) = \gamma_{\text{peak}} \tag{B.17b}$$
$$\mathrm{ACLR}(\boldsymbol{\theta}) \leq \beta_{\text{leak}} \tag{B.17c}$$

où γ_{peak} et β_{leak} désignent respectivement le PAPR et l'ACLR cibles. Notez que le PAPR et l'ACLR dépendent des paramètres de l'émetteur $\boldsymbol{\theta}$.

L'un des principaux avantages d'une implémentation de la paire émetteur-récepteur en tant que système E2E est qu'elle permet l'optimisation des paramètres entraînables par descente de gradient stochastique (stochastic gradient descent, SGD). Cela nécessite une fonction de

perte différentiable afin que les gradients puissent être calculés et rétro-propagés à travers le système E2E. Dans ce qui suit, nous exprimons donc l'objectif (B.17a) et les contraintes (B.17b) et (B.17c) comme des fonctions différentiables qui peuvent être évaluées pendant l'apprentissage et minimisées avec la SGD. Tout d'abord, le taux réalisable (B.17a) peut être exprimé de manière équivalente en utilisant le système d'entropie croisée binaire (binary cross-entropy, BCE) [21], qui est souvent utilisé dans ce genre de problèmes de classification binaire :

$$L_C(\boldsymbol{\theta}, \boldsymbol{\psi}) := -\frac{1}{MN} \sum_{m\in\mathcal{M}} \sum_{n\in\mathcal{N}} \sum_{q=0}^{Q-1} \mathbb{E}_{\mathbf{y}_m} \left[\log_2 \left(\widehat{P}_{\boldsymbol{\psi}}(b_{m,n,q}|\mathbf{y}_m) \right) \right] \tag{B.18}$$

$$= Q - C(\boldsymbol{\theta}, \boldsymbol{\psi}). \tag{B.19}$$

Pour surmonter la complexité associée au calcul de la valeur attendue, une approximation est généralement obtenue par un échantillonnage de Monte Carlo :

$$L_C(\boldsymbol{\theta}, \boldsymbol{\psi}) \approx -\frac{1}{MNB_S} \sum_{m\in\mathcal{M}} \sum_{n\in\mathcal{N}} \sum_{q=0}^{Q-1} \sum_{i=0}^{B_S-1} \log_2 \left(\widehat{P}_{\boldsymbol{\psi}} \left(b^{[i]}_{m,n,q} | \mathbf{y}^{[i]}_m \right) \right). \tag{B.20}$$

Deuxièmement, l'évaluation de la contrainte (B.17b) nécessite le calcul de la probabilité $P(\frac{|s(t)|^2}{\mathbb{E}[|s(t)|^2]} > e)$, où e est le seuil d'énergie défini dans (B.10). Cependant, le calcul de cette probabilité est d'une complexité prohibitive en raison du grand nombre de symboles OFDM possibles. Pendant l'apprentissage, nous appliquons donc cette contrainte avec $\epsilon = 0$ et en pénalisant tous les signaux dont l'amplitude au carré dépasse γ_{peak}. Avec $\epsilon = 0$, la contrainte (B.17b) est équivalente à imposer $L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) = 0$, avec

$$L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) = \mathbb{E}_m \left[\int_{-\frac{T}{2}}^{\frac{T}{2}} \left(|s_m(t)|^2 - \gamma_{\text{peak}} \right)^+ dt \right] \tag{B.21}$$

où $(x)^+$ désigne la partie positive de x, c'est-à-dire $(x)^+ = \max(0, x)$. Pour évaluer $L_{\gamma_{\text{peak}}}(\boldsymbol{\theta})$ pendant l'entraînement, la valeur de l'espérance peut être obtenue par échantillonnage de Monte Carlo, et l'intégrale peut être approximée en utilisant une somme de Riemann :

$$L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) \approx \frac{T}{B_S NO_S} \sum_{i=0}^{B_S-1} \sum_{t=-\frac{NO_S-1}{2}}^{\frac{NO_S-1}{2}} \left(\left| \underline{z}^{[i]}_{m,t} \right|^2 - \gamma_{\text{peak}} \right)^+ \tag{B.22}$$

où $\underline{\mathbf{z}}_m = \mathbf{F}^{\mathsf{H}} \mathbf{x}_m \in \mathbb{C}^{NO_S}$ est le vecteur du signal temporel suréchantillonné correspondant à la sortie $\mathbf{x}_m$ de l'émetteur neuronal, avec $\mathbf{F}^{\mathsf{H}} \in \mathbb{C}^{NO_S \times N}$ la matrice d'IDFT.

Troisièmement, la contrainte d'inégalité (B.17c) peut être convertie en une contrainte d'égalité $\text{ACLR}(\boldsymbol{\theta}) - \beta_{\text{leak}} = -v$, où $v \in \mathbb{R}_+$ est une variable libre (*slack variable*). Cette

contrainte d'égalité est alors appliquée en minimisant $L_{\beta_{\text{leak}}}(\boldsymbol{\theta}) + v$, avec

$$L_{\beta_{\text{leak}}}(\boldsymbol{\theta}) = \frac{\mathbb{E}\left[E_A\right]}{\mathbb{E}\left[E_I\right]} - 1 - \beta_{\text{leak}} \tag{B.23}$$

$$\approx \frac{\frac{1}{B_S}\sum_{i=0}^{B_S-1} \mathbf{x}^{[i]\mathsf{H}}\mathbf{W}\mathbf{x}^{[i]}}{\frac{1}{B_S}\sum_{i=0}^{B_S-1} \mathbf{x}^{[i]\mathsf{H}}\mathbf{V}\mathbf{x}^{[i]}} - 1 - \beta_{\text{leak}}. \tag{B.24}$$

Enfin, pour $\epsilon = 0$, le problème (B.17) peut être reformulé comme suit

$$\underset{\boldsymbol{\theta},\boldsymbol{\psi}}{\text{minimize}} \quad L_C(\boldsymbol{\theta},\boldsymbol{\psi}) \tag{B.25a}$$

$$\text{subject to} \quad L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) = 0 \tag{B.25b}$$

$$L_{\beta_{\text{leak}}}(\boldsymbol{\theta}) + v = 0 \tag{B.25c}$$

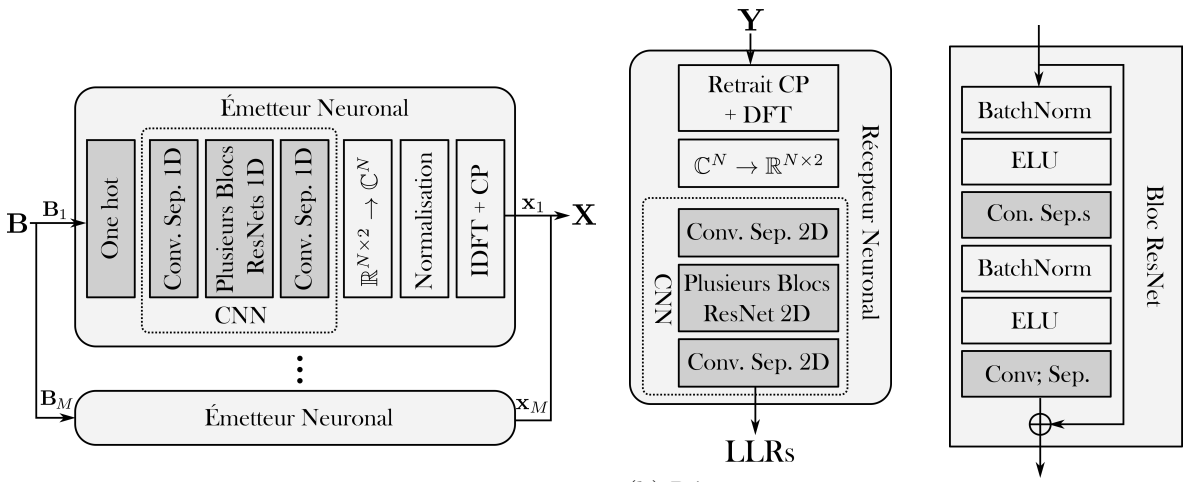
où l'objectif et les contraintes sont différentiables et peuvent être estimés lors de l'entraînement. Pour entraîner le système de bout en bout, nous utilisons ensuite la méthode du Lagrangien augmentée, qui utilise deux types d'hyperparamètres qui sont mis à jour de manière itérative pendant l'entraînement. Désignons par $\mu_p > 0$ et $\mu_l > 0$ les paramètres de pénalité et par λ_p et λ_l les multiplicateurs de Lagrange pour les fonctions de contrainte $L_{\gamma_{\text{peak}}}(\boldsymbol{\theta})$ et $L_{\beta_{\text{leak}}}(\boldsymbol{\theta})$, respectivement. Le Lagrangien augmenté correspondant est défini comme suit [79]

$$\begin{aligned}\overline{L}(\boldsymbol{\theta},\boldsymbol{\psi},\lambda_p,\lambda_l,&\mu_p,\mu_l) = L_C(\boldsymbol{\theta},\boldsymbol{\psi}) \\ &+ \lambda_p L_{\gamma_{\text{peak}}}(\boldsymbol{\theta}) + \frac{1}{2}\mu_p |L_{\gamma_{\text{peak}}}(\boldsymbol{\theta})|^2 \\ &+ \frac{1}{2\mu_l}\left(\max(0, \lambda_l + \mu_l L_{\beta_{\text{leak}}}(\boldsymbol{\theta}))^2 - \lambda_l^2\right)\end{aligned} \tag{B.26}$$

où la minimisation par rapport à v a été effectuée pour chaque paire fixe de $\{\boldsymbol{\theta},\boldsymbol{\psi}\}$ [79]. Chaque itération d'apprentissage comprend de multiples étapes de SGD sur le Lagrangien augmenté (B.26) suivies d'une mise à jour des hyperparamètres. La procédure d'optimisation est détaillée dans l'Algorithme 2 présent dans la Section 4.3.1.

Architecture du système

L'émetteur et le récepteur neuronaux sont basés sur des architectures similaires, représentées schématiquement dans la Figure B.10. L'élément central est un bloc *ResNet*, qui est constitué de deux séquences identiques de couches suivies de l'ajout de l'entrée, comme illustré dans la Figure B.10c. Chaque séquence est composée d'une couche de normalisation par batch, d'une fonction d'activation ELU et d'une convolution séparable. Les architectures précises de l'émetteur et du récepteur basés sur des CNNs sont détaillées dans la Section 4.3.2. Notez qu'aucun pilote n'est utilisé car il a été démontré dans [16] que la communication sans pilote est possible sur les canaux OFDM lorsque des récepteurs neuronaux sont utilisés.



(a) Transmetteur neuronal.

(b) Récepteur neuronal.

(c) Bloc ResNet.

FIGURE B.10 : Différentes parties du système de bout en bout, où les blocs grisés sont des composants entraînables.

B.3.4 Résultats des Simulations

Des ensembles de données distincts ont été utilisés pour l'entraînement et le test, tous deux générés à l'aide d'un mélange de modèles à visibilité directe (line-of-sight, LOS) et à visibilité indirecte (non-LOS) conformes aux normes 3GPP-UMi (urban microcell). Nous avons supposé un contrôle parfait de la puissance, de sorte que l'énergie moyenne du canal par RE était de un, c'est-à-dire $\mathbb{E}\left[|h_{m,n}|^2\right] = 1$. Le système considéré comporte $N = 75$ sous-porteuses avec $M = 14$ symboles OFDM en utilisant des CPs de longueurs suffisantes pour que le canal puisse être représenté par (B.6) dans le domaine fréquentiel. La fréquence centrale de la porteuse et l'espacement entre les sous-porteuses ont été fixés respectivement à 3.5 GHz et 30 kHz, et le nombre de bits par utilisation du canal a été fixé à $Q = 4$. Les comparaisons du taux d'erreur sur les bits codés (coded BER) ont été effectuées à l'aide d'un code LDPC (low-density parity-check) standard conforme à la norme 5G, d'une longueur de 1024 et de taux de codage $\eta = \frac{1}{2}$. Le système TR de référence a été évalué pour $R \in \{0, 2, 4, 8, 16, 32\}$ PRT et le système E2E a été entraîné pour atteindre des ACLRs de $\beta_{\text{leak}} \in \{-20, -30, -40\}$ dB et des PAPRs de $\gamma_{\text{peak}} \in \{4, 5, 6, 7, 8, 9\}$ dB. Finallement, le SNR $= \frac{\mathbb{E}_{h_{m,n}}\left[|h_{m,n}|^2\right]}{\sigma^2} = \frac{1}{\sigma^2}$ a été choisi au hasard dans l'intervalle $[10, 30]$ dB pour chaque RG du batch pendant l'entraînement.

Les taux de transmission moyens par RE obtenus par le système de référence et les systèmes E2E sont présentés dans la Figure B.11 où les chiffres à côté des points de données sont les ACLR correspondants. Tout d'abord, on peut voir qu'au PAPR maximal d'environ 8.5 dB, le système E2E entraîné avec $\beta_{\text{leak}} = -20$ dB atteint un débit supérieur de 3% à celui du système de référence sans PRTs. Cela peut s'expliquer par la perte de débit due à la présence de pilotes dans le système de référence, qui ne transportent pas de données et représentent environ 4% du nombre total de REs. Ensuite, à des PAPR plus faibles, les taux de transmisison obtenus

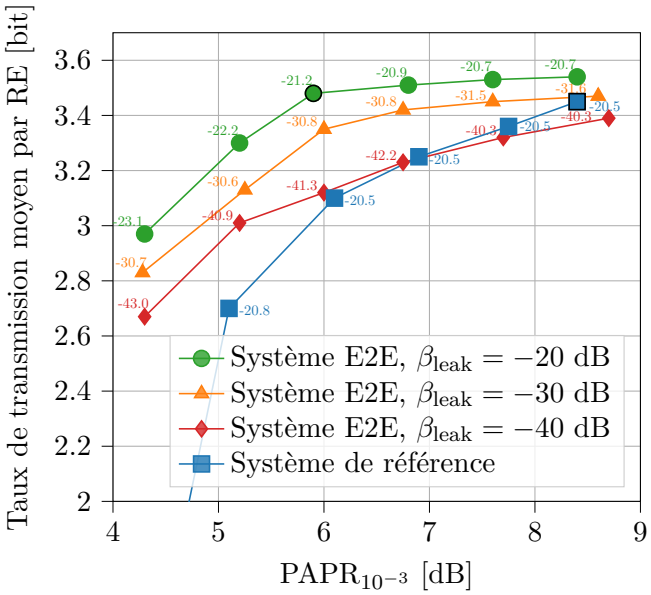


FIGURE B.11 : Taux de transmission obtenus pour les systèmes comparés. Les nombres près des points de données indiquent les ACLRs correspondants.

par le système E2E entraîné avec $\beta_{\text{leak}} \in \{-20, -30\}$ sont significativement plus élevés que ceux obtenus par le système de référence. Enfin, les systèmes E2E sont capables d'atteindre leurs objectifs respectifs de PAPR et d'ACLR.

Comme le système de référence transmet des pilotes et des signaux de réduction en plus des signaux de données, l'énergie par bit transmis est plus élevée que celle des systèmes E2E. Pour refléter cette caractéristique, nous définissons le rapport énergie par bit sur densité spectrale du bruit comme suit

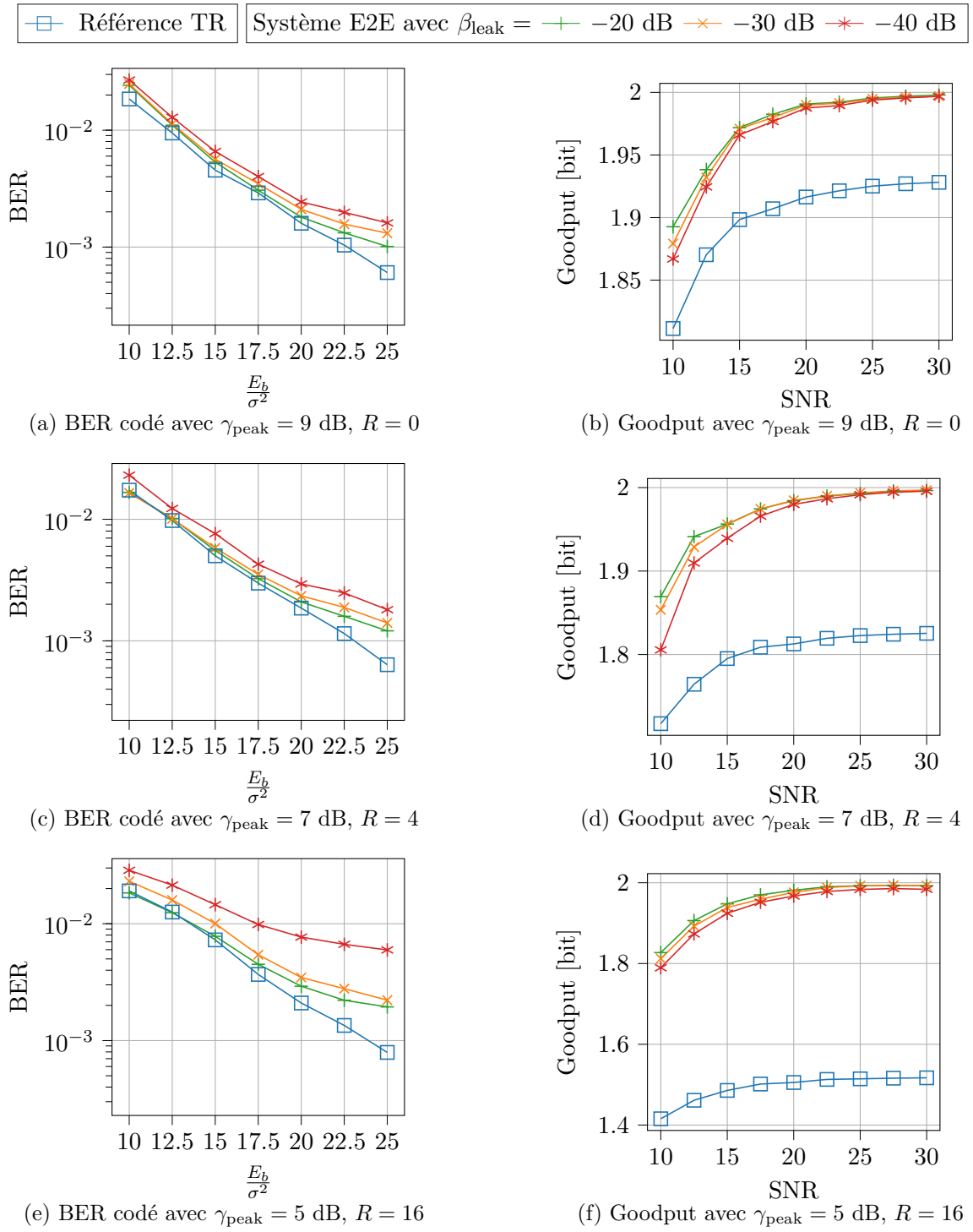
$$\frac{E_b}{\sigma^2} = \frac{\mathbb{E}_{x_{m,n}}\left[|x_{m,n}|^2\right]}{\rho K \sigma^2} = \frac{1}{\rho Q \sigma^2} \tag{B.27}$$

où ρ est le rapport entre le nombre de RE transportant des signaux de données et le nombre total de REs dans la RG. Les BER codés du système de référence et des trois systèmes E2E sont présentés dans la colonne de gauche de la Figure B.12. On peut voir que le système de base obtient systématiquement un BER légèrement inférieur à celui des systèmes E2E. Ceci est attendu car le système de base transmet également moins de bits par RG, du fait que certains REs sont utilisés pour transmettre des pilotes ou des signaux de réduction.

Pour comprendre les avantages offerts par l'approche E2E, la deuxième colonne de la Figure B.12 présente les *goodputs* obtenus par chaque système comparé. Le goodput est défini comme le nombre moyen de bits d'information qui ont été reçus avec succès dans un RE :

$$\text{Goodput} = \eta \rho Q (1 - \text{BER}), \tag{B.28}$$

Les résultats de l'évaluation montrent que les goodputs obtenus par tous les systèmes entraînés,



(a) BER codé avec $\gamma_{\text{peak}} = 9$ dB, $R = 0$

(b) Goodput avec $\gamma_{\text{peak}} = 9$ dB, $R = 0$

(c) BER codé avec $\gamma_{\text{peak}} = 7$ dB, $R = 4$

(d) Goodput avec $\gamma_{\text{peak}} = 7$ dB, $R = 4$

(e) BER codé avec $\gamma_{\text{peak}} = 5$ dB, $R = 16$

(f) Goodput avec $\gamma_{\text{peak}} = 5$ dB, $R = 16$

FIGURE B.12 : BERs et goodputs obtenus par les différents systèmes.

y compris ceux entraînés pour des ACLR plus faibles, sont nettement supérieurs à ceux du système de référence. Ces gains sont rendus possibles par le fait que le système E2E n'utilise pas de pilotes, par le schéma efficace de réduction de l'ACLR appris par la procédure d'optimisation proposée, et par le fait que tous les REs peuvent être utilisés pour transmettre des données.

Des observations d'évaluation disponibles dans la Section 4.4.2 révèlent que l'émetteur neuronal parvient à une réduction du PAPR et de l'ACLR grâce à un filtrage dépendant de la sous-porteuse, une distribution inégale de l'énergie sur la sous-porteuse et un réajustement de la position de chaque point dans la constellation. D'autre part, le récepteur neuronal est capable d'égaliser le canal sans aucun pilote transmis grâce aux constellations asymétriques apprises. Cela se traduit directement par des gains de débit puisque les transmissions additionnelles associées à l'envoi des pilotes sont supprimée.

B.3.5 Conclusion

Dans ce chapitre, nous avons proposé une approche d'apprentissage pour concevoir des formes d'onde OFDM qui répondent à des contraintes spécifiques sur l'enveloppe et les caractéristiques spectrales. Nous avons exploité la stratégie de bout en bout pour modéliser l'émetteur et le récepteur comme deux CNNs qui effectuent respectivement une modulation et une démodulation de grande dimension. La procédure d'apprentissage associée exige d'abord que toutes les contraintes d'optimisation soient exprimées sous forme de fonctions différentiables qui peuvent être minimisées par SGD. Ensuite, un problème d'optimisation sous contrainte est formulé et résolu à l'aide de la méthode du Lagrangien augmenté. Les résultats des simulations montrent que la procédure d'optimisation permet de concevoir des formes d'onde qui satisfont des contraintes de PAPR et d'ACLR. De plus, le système de bout en bout permet un débit jusqu'à 30% plus élevé qu'une implémentation quasi optimale d'un système TR avec un ACLR et un PAPR similaires. Il est intéressant de noter que ces améliorations sont possibles parce que le système de communication est entièrement conçu grâce au DL. Les normes actuelles exigent toutefois l'utilisation de modulations et de placement de pilotes bien connus afin de garantir la compatibilité avec une large gamme de matériel. Les implémentations à court terme de systèmes de bout en bout sont donc peu probables, et des solutions plus pratiques doivent être trouvées pour libérer le potentiel du DL dans un avenir proche.

B.4 Récepteur Amélioré par DL pour les Systèmes MU-MIMO

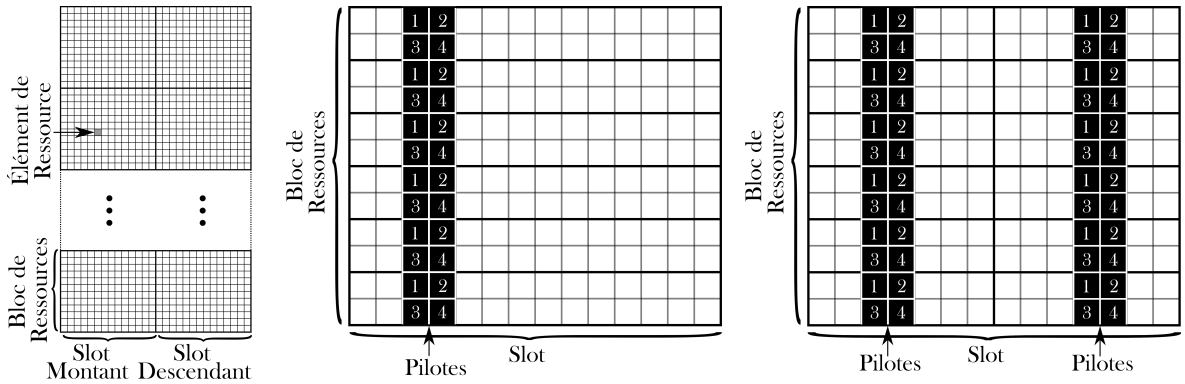
B.4.1 Motivations

Les stratégies d'optimisation basées sur les blocs et de bout en bout présentent toutes deux des avantages et des inconvénients qui ont été examinés respectivement dans les Sections B.2 et B.3. D'une part, la stratégie basée sur les blocs pour la détection MU-MIMO a l'avantage de rester interprétable et adaptable à un nombre variable d'utilisateurs, mais elle n'est pas entraînée à optimiser les performances de bout en bout. D'autre part, la stratégie de bout en bout permet l'émergence de formes d'onde profondément optimisées, mais les émetteurs-récepteurs basés sur des NNs sont coûteux en calcul et ne sont pas adaptés aux transmissions MU-MIMO. Comme nous l'avons vu dans la Section B.2, la plupart des travaux traitant de la détection MU-MIMO améliorée par DL ne prennent en compte que l'étape d'égalisation. Cependant, il a été proposé dans [82] de modéliser tout un récepteur à utilisateur unique MIMO (SU-MIMO) comme un seul NN grâce à une couche dite 'de transformation'. Cette solution, appelée DeepRX MIMO, présente des gains importants mais reste très coûteuse en termes de calcul. Les principaux inconvénients de ces récepteurs MIMO basés sur des NNs restent leur manque d'interprétabilité et d'adaptabilité à un nombre variable d'utilisateurs.

Dans ce contexte, nous avons introduit une nouvelle approche hybride dans laquelle plusieurs composants DL sont entraînés de bout en bout pour améliorer un récepteur MU-MIMO classique [83]. L'objectif est de combiner l'interprétabilité de la première stratégie, l'efficacité de la seconde et la flexibilité des récepteurs traditionnels. Notre solution améliore la prédiction des statistiques d'erreur d'estimation du canal et utilise un démappeur basé sur un CNN pour affiner les estimations de la probabilité des bits. L'architecture proposée est évaluée sur des modèles de canaux conformes à la norme 3GPP pour les transmissions en liaison montante et descendante. Deux systèmes de références ont été implémentés, le premier étant un récepteur conventionnel, et le second étant le même récepteur mais avec une connaissance parfaite du canal aux positions pilotes et des statistiques d'erreur d'estimation du canal. Nos résultats montrent que les gains apportés par le récepteur amélioré par DL augmentent avec la vitesse de l'utilisateur, avec de petites améliorations du BER à faible vitesse mais des améliorations significatives à grande vitesse.

B.4.2 Modélisation du Système

Nous considérons un système MU-MIMO, dans lequel K utilisateurs à antenne unique communiquent avec une BS équipée de L antennes sur les liaisons montante et descendante. On considère des transmissions OFDM, et la RG est divisé en blocs de ressources constitués de 12 sous-porteuses adjacentes (Figure B.13a). Des modulations d'amplitude en quadrature (quadrature amplitude modulation, QAM) d'ordre 2^Q sont utilisées pour transmettre les données. Les coefficients de canal forment un tenseur à 4 dimensions noté $\mathbf{H} \in \mathbb{C}^{2M \times N \times L \times K}$, tel que $\mathbf{H}_{m,n} \in \mathbb{C}^{L \times K}$ est la matrice de canal pour le RE (m, n), et $\mathbf{h}_{m,n,k} \in \mathbb{C}^{L}$ est le vecteur de canal pour le RE (m, n) et pour l'utilisateur k. Le duplexage est réalisé par division dans le



(a) Grille de ressources.

(b) Configuration de pilotes 1P pour $K = 4$ utilisateurs.

(c) Configuration de pilotes 2P pour $K = 4$ utilisateurs.

FIGURE B.13 : Les pilotes sont disposés sur la RG selon deux configurations différentes, où chaque numéro correspond à un émetteur différent.

temps (time division duplex, TDD), de sorte qu'un slot est attribué alternativement à la liaison montante ou à la liaison descendante, comme l'illustre la Figure B.13B. Deux configurations de pilotes différentes sont considérées, appelées configurations de pilotes 1P et 2P, et contiennent respectivement des pilotes sur deux ou quatre symboles OFDM dans un slot. Les Figures B.13b et B.13c montrent respectivement les configurations de pilotes 1P et 2P pour 4 utilisateurs.

L'ensemble des REs portant les pilotes correspondant à un utilisateur $k \in \{1, \ldots, K\}$ est désigné par $\mathcal{P}^{(k)}$ et les nombres de symboles et de sous-porteuses portant les pilotes sont respectivement désignés par $|\mathcal{P}_M|$ et $|\mathcal{P}_N|$. A titre d'exemple, si la configuration de pilotes 1P représentée dans la Figure B.13b est utilisée avec $N = 12$, les positions $(\text{symboles}, \text{sous-porteuses})$ de tous les REs portant des pilotes pour l'utilisateur 1 sont désignées par $\mathcal{P}^{(1)} = \{(3,1), (3,3), (3,5), (3,73), (3,93), (3,11)\}$, ce qui donne $|\mathcal{P}_M| = 1$ et $|\mathcal{P}_N| = 6$. On notera que les pilotes ne subissent aucune interférence. La puissance du bruit est désignée par σ^2 et est supposée égale pour tous les utilisateurs et tous les REs. Dans ce qui suit, nous supposons un contrôle parfait de la puissance sur la RG de sorte que l'énergie moyenne correspondant à une seule antenne de BS et à un seul utilisateur est égale à un, c'est-à-dire, $\mathbb{E}\left[|h_{m,n,l,k}|^2\right] = 1$. Le rapport signal/bruit de la transmission est défini comme suit

$$\text{SNR} = 10 \log \left(\frac{\mathbb{E}\left[|h_{m,n,l,k}|^2\right]}{\sigma^2} \right) = 10 \log \left(\frac{1}{\sigma^2} \right) \text{[dB]}. \tag{B.29}$$

En liaison montante, la BS vise à récupérer les bits transmis simultanément par les K utilisateurs sur les REs transportant les données. Les tenseurs des signaux émis et reçus de tous les utilisateurs sont respectivement notés $\mathbf{X} \in \mathbb{C}^{2M \times N \times K}$ et $\mathbf{Y} \in \mathbb{C}^{2M \times N \times L}$, et la fonction de transfert sur la liaison montante est $\mathbf{y}_{m,n} = \mathbf{H}_{m,n}\mathbf{x}_{m,n} + \mathbf{n}_{m,n}$, où $\mathbf{n}_{m,n} \sim \mathcal{CN}\left(\mathbf{0}, \sigma^2 \mathbf{I}_L\right)$ est le vecteur de bruit. Dans ce scénario , seul le créneau de liaison montante est utilisé et, par conséquent, tous les signaux ayant des indices $m > M$ sont ignorés, c'est-à-dire que les valeurs correspondantes sont fixées à 0. L'architecture du système en liaison montante est illustrée

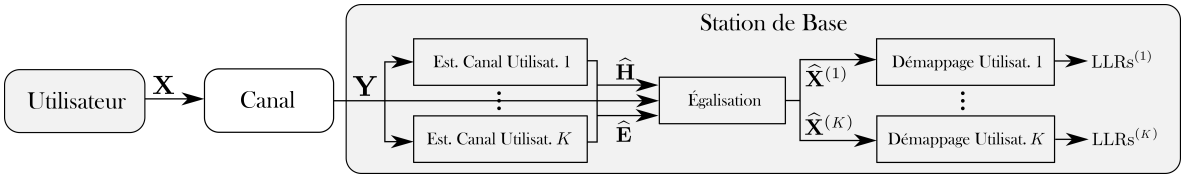


FIGURE B.14 : Architecture du système de communication pour la liaison montante.

dans la Figure B.14, où une IDFT (DFT) et l'ajout (suppression) du préfixe cyclique avant (après) le canal ne sont pas représentés pour plus de clarté. Les étapes d'estimation du canal, d'égalisation et de démappage du système de référence sont expliquées dans ce qui suit.

Estimation du canal pour la liaison montante

Comme les pilotes sont supposés être orthogonaux, l'estimation du canal LMMSE peut être effectuée pour chaque utilisateur indépendamment. La matrice de covariance du canal fournissant les corrélations spatiales, temporelles et spectrales entre tous les REs portant des pilotes est notée $\mathbf{\Sigma} \in \mathbb{C}^{|\mathcal{P}_M|\cdot|\mathcal{P}_N|\cdot L \times |\mathcal{P}_m|\cdot|\mathcal{P}_n|\cdot L}$. Pour un utilisateur $k \in \{1, \ldots, K\}$, l'estimation du canal LMMSE au niveau des REs portant des pilotes est notée par $\widehat{\mathbf{H}}^{(k)}_{\mathcal{P}^{(k)}} \in \mathbb{C}^{|\mathcal{P}_M|\times|\mathcal{P}_N|\times L}$. Inspiré par les directives 3GPP [91], les estimations de canal pour tous les REs sont calculées en interpolant d'abord linéairement les estimations des REs portant des pilotes dans la dimension fréquentielle, puis en utilisant l'estimation au RE interpolé le plus proche (nearest interpolated resource element, NIRE) sur les REs voisins.

Il est également possible d'exploiter l'interpolation linéaire temporelle entre les symboles OFDM portant les pilotes lorsque la configuration de pilotes 2P est utilisée. Le tenseur d'estimations de canal ainsi obtenu est noté $\widehat{\mathbf{H}}^{(k)} \in \mathbb{C}^{2M\times N\times L}$. L'estimation globale du canal pour tous les utilisateurs $\widehat{\mathbf{H}} \in \mathbb{C}^{2M\times N\times L\times K}$ est obtenue en concaténant les estimations de canal de tous les utilisateurs. Puisque seul le slot de liaison montante est considéré ici, les estimations de canal pour les M derniers symboles (slot de liaison descendante) sont nulles. L'erreur d'estimation du canal est notée $\widetilde{\mathbf{H}}$ et est telle que $\mathbf{H} = \widehat{\mathbf{H}} + \widetilde{\mathbf{H}}$. Pour un RE (m, n), nous définissons

$$\mathbf{E}_{m,n} \coloneqq \mathbb{E}\left[\widetilde{\mathbf{H}}_{m,n}\widetilde{\mathbf{H}}^{\mathsf{H}}_{m,n}\right] \tag{B.30}$$

comme la somme des matrices de covariance des erreurs d'estimation *spatiales* du canal de tous les utilisateurs. L'estimation de ces matrices de covariance, utilisant une approximation NIRE similaire, est détaillée dans la Section 5.4.2.

Égalisation de la liaison montante

Nous utilisons l'égaliseur LMMSE, très répandu car il maximise le rapport signal/bruit après égalisation. Cependant, étant donné que le calcul d'un opérateur LMMSE dédié pour chaque RE est irréalisable en pratique en raison de sa complexité prohibitive, nous avons recours à un égaliseur LMMSE groupé, c'est-à-dire qu'un seul opérateur LMMSE est appliqué à un groupe de REs adjacents couvrant plusieurs symboles et sous-porteuses. Les symboles

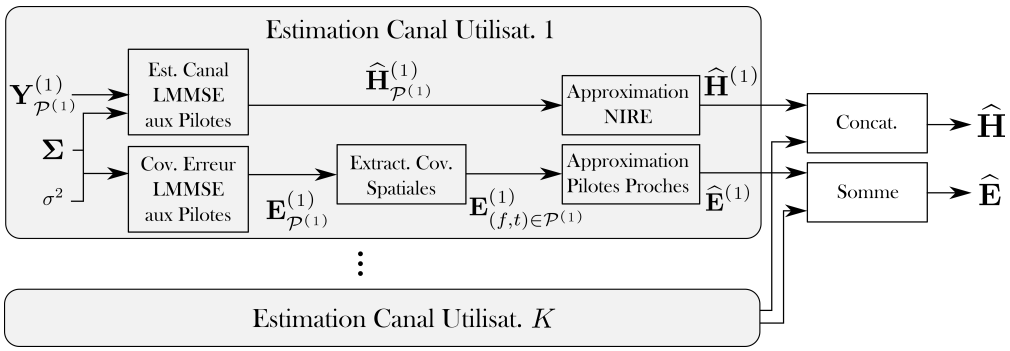


FIGURE B.15 : Estimation du canal pour la liaison montante.

égalisés de l'utilisateur k sont notés $\widehat{\mathbf{X}}^{(k)} \in \mathbb{C}^{M \times N}$, comme illusté dans la Figure B.14. Une dérivation de l'estimateur LMMSE groupé est donnée en Appendix A.

Démappage de la liaison montante

Après égalisation, le canal de liaison montante peut être considéré comme MNK canaux de bruit additif parallèles qui peuvent être démodulés indépendamment pour chaque RE et chaque utilisateur. Pour un RE (m, n) et un utilisateur k, le canal post-égalisation s'exprime par $\hat{x}_{m,n,k} = x_{m,n,k} + \zeta_{m,n,k}$, où le bruit $\zeta_{m,n,k}$ comprend à la fois les interférences et le bruit subi par l'utilisateur k. Sa variance est donnée par $\rho^2_{m,n,k} = \mathbb{E}\left[\zeta^*_{m,n,k}\zeta_{m,n,k}\right]$, et son calcul est donné dans la Section 5.2.2. Un démappage standard pour un bruit blanc additif gaussien (additive white Gaussian noise, AWGN) est finalement effectué sur chaque symbole $\hat{x}_{m,n,k}$ en utilisant sa variance de bruit correspondante $\rho^2_{m,n,k}$.

Système de référence pour la liaison descendante

Dans la liaison descendante, la BS doit transmettre simultanément à K utilisateurs sur tous les REs du slot descendant. Désignons par $\mathbf{S} \in \mathcal{C}^{2M \times N \times K}$ et par $\mathbf{T} \in \mathbb{C}^{2M \times N \times L}$ les tenseurs des symboles non précodés et précodés, respectivement. Nous désignons par $\mathbf{U} \in \mathbb{C}^{2M \times N \times K}$ le tenseur des symboles reçus par les K utilisateurs. Ces quantités ne sont pertinentes que sur le slot descendant et sont donc considérées comme nulles sur les M premiers symboles. La fonction de transfert du canal en liaison descendante pour un RE (m, n) est

$$\mathbf{u}_{m,n} = \mathbf{H}^{\mathsf{H}}_{m,n}\mathbf{t}_{m,n} + \mathbf{q}_{m,n} \tag{B.31}$$

où $\mathbf{q}_{m,n} \sim \mathcal{CN}(\mathbf{0}, \sigma^2\mathbf{I}_K)$ est le vecteur de bruit, considéré nul sur les M premiers symboles. Les étapes de précodage, d'estimation et d'égalisation des canaux et de démappage de la liaison descendante sont détaillées dans la Section 5.2.3.

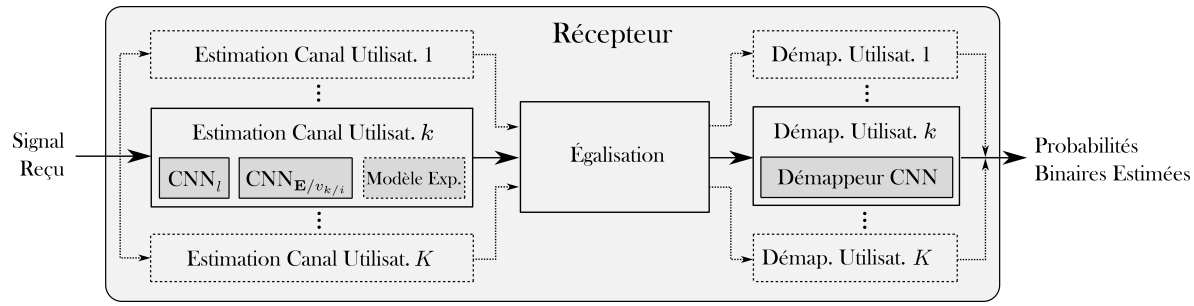


FIGURE B.16 : Architecture de récepteur améliorée par DL. Les éléments en pointillés ne sont présents que sur la liaison montante, où la station de base traite tous les utilisateurs.

B.4.3 Architecture du Récepteur Amélioré par DL

Les systèmes de référence présentés dans la section précédente présentent plusieurs limites. En particulier, l'approximation NIRE conduit à des erreurs d'estimation de canal élevées pour les REs qui sont éloignés des pilotes. De même, l'égalisation groupée peut être inexacte pour ces REs. Cette section détaille l'architecture d'un récepteur qui s'appuie sur le système de référence mais utilise plusieurs CNNs pour améliorer ses performances.

Entraînement du récepteur

L'architecture du récepteur amélioré par DL est illustrée dans la Figure B.16, où les composants entraînables sont représentés en gris foncé. Dans la liaison descendante, chaque utilisateur k effectue uniquement l'estimation de canal, l'égalisation et le démappage de son propre signal, et les composants correspondantes sont illustrées par des contours continus. En revanche, sur la liaison montante, la station de base traite tous les utilisateurs en parallèle, et les composants supplémentaires sont délimitées par des lignes pointillées. L'adaptabilité au nombre d'utilisateurs est obtenue en utilisant différentes copies des mêmes composants DL pour chaque utilisateur, toutes les copies partageant le même ensemble de paramètres entraînables. Nous proposons d'optimiser conjointement tous ces composants en se basant uniquement sur les probabilités binaires estimées, et non en entraînant chacun d'entre eux individuellement. Cette approche est pratique car elle ne suppose pas la connaissance des coefficients du canal lors de l'entraînement. Désignons par $\boldsymbol{\theta}$ l'ensemble des paramètres entraînables du récepteur amélioré par DL, et par $b_{m,n,k,q}$ le bit (m, n, k, q) envoyé. Ces paramètres sont optimisés pour minimiser la BCE totale :

$$\mathcal{L} \triangleq -\sum_{k=1}^{K} \sum_{(m,n)\in\mathcal{D}} \sum_{q=0}^{Q-1} \mathbb{E}_{b,\mathbf{Y}} \left[\log_2 \left(\widehat{P}_{\boldsymbol{\theta}} \left(b_{m,n,k,q} | \mathbf{Y} \right) \right) \right] \tag{B.32}$$

où $\mathcal{D}$ désigne l'ensemble des REs transportant des données et $\widehat{P}_{\boldsymbol{\theta}}(\cdot|\mathbf{Y})$ est l'estimation du récepteur de distribution postérieure sur les bits sachant $\mathbf{Y}$. L'espérance dans (B.32) est

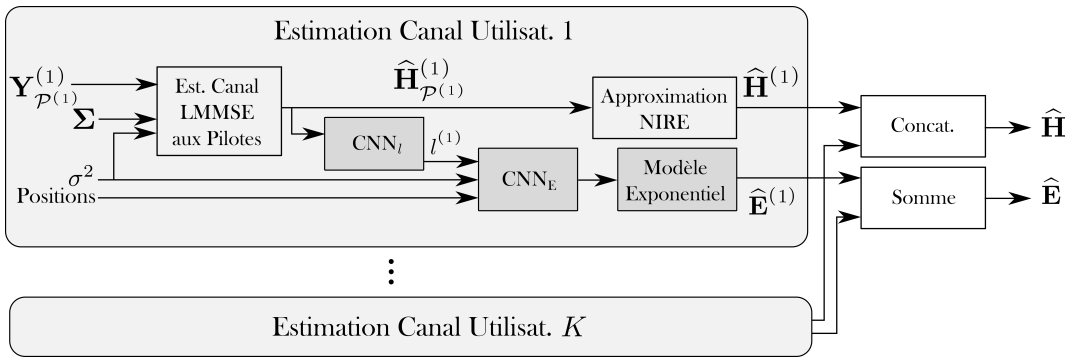


FIGURE B.17 : Estimation du canal de la liaison montante améliorée par DL.

estimée par échantillonnage de Monte Carlo en utilisant des batchs contenant B_S examples :

$$\mathcal{L} \approx -\frac{1}{B_S}\sum_{i=1}^{B_S}\sum_{k=1}^{K}\sum_{(m,n)\in\mathcal{D}}\sum_{q=0}^{Q-1}\left(\log_2\left(\widehat{P}_{\boldsymbol{\theta}}\left(b^{[i]}_{m,n,k,q}|\mathbf{Y}^{[i]}\right)\right)\right) \tag{B.33}$$

où l'exposant $[i]$ désigne le $i^{\text{ième}}$ example dans le batch. En suivant une dérivation similaire à celle proposée dans la Section 2.3.3, la perte (B.32) peut être redéfinie comme

$$\mathcal{L} = \sum_{k=1}^{K}\left(\mathrm{Card}(\mathcal{D})Q - C_k\right) \tag{B.34}$$

où $\mathrm{Card}(\mathcal{D})$ est le nombre de REs transporant des données, $\mathrm{Card}(\mathcal{D})Q$ est le nombre de bits transmis par un utilisateur, et C_k est un taux de transmission atteignable pour l'utilisateur k.

Estimateur de canal amélioré par DL

Dans le récepteur conventionnel, les statistiques d'erreur d'estimation du canal ne peuvent être obtenues que pour les REs qui transportent des pilotes, ce qui signifie que la précision de l'estimation diminue à mesure que l'on s'en éloigne. Dans ce qui suit, nous présentons des CNNs qui estiment les matrices de covariance des erreurs d'estimation du canal sur la liaison montante et les variances des erreurs d'estimation sur la liaison descendante. Sur la liaison montante (Figure B.17), les matrices de covariance des erreurs d'estimation spatiales du canal $\mathbf{E}_{m,n}$ sont nécessaires pour calculer les symboles égalisés et la variance du bruit post-égalisation. Pour prédire chaque élément de $\widehat{\mathbf{E}}^{(k)}$ pour l'utilisateur k, un CNN conçu naïvement produirait MNL^2 paramètres complexes. Cela serait d'une complexité prohibitive pour tout grand nombre de sous-porteuses, de symboles ou d'antennes de réception. Pour cette raison, nous approximons chaque élément (x, y) de $\widehat{\mathbf{E}}^{(k)}_{m,n}$ avec le modèle exponentiel suivant :

$$\hat{e}^{(k)}_{m,n,a,b} = \alpha_{m,n}\beta^{|b-a|}_{m,n}\exp\left(j\gamma(b-a)\right) \tag{B.35}$$

où $b-a$ est la différence de position horizontale entre cet élément et la diagonale, et $\alpha_{m,n}$, $\beta_{m,n}$, et γ sont des paramètres de ce modèle. Les paramètres $\alpha_{m,n}$ et $\beta_{m,n}$ contrôlent respectivement l'amplitude et la décroissance du modèle, et dépendent du RE (m, n). Pour estimer ces deux paramètres pour chaque RE, nous utilisons un CNN, désigné par $\text{CNN}_{\mathbf{E}}$, qui admet quatre entrées, chacune de taille $M \times N$, pour une dimension d'entrée totale de $M \times N \times 4$. Les deux premières entrées fournissent l'emplacement de chaque RE dans la RG, et la troisième entrée fournit le SNR de la transmission. Enfin, la quatrième entrée est une caractéristique $l^{(k)} \in \mathbb{R}$ fournie par un autre CNN, désigné par CNN_l, qui a été conçu avec l'intuition de prédire la variabilité temporelle du canal expérimentée par l'utilisateur k. Pour ce faire, CNN_l utilise les estimations du canal au niveau des REs portant des pilotes pour estimer l'étalement Doppler. Il produit le scalaire $l^{(k)}$, qui est transmis à $\text{CNN}_{\mathbf{E}}$ sous forme de matrice $l^{(k)} \cdot \mathbb{1}_{M\times N}$. De plus amples détails sont donnés dans la Section 5.3.2.

Dans la liaison descendante, les variances des erreurs d'estimation des canaux sont estimées de manière similaire. Cependant, chaque utilisateur estime à la fois son propre canal et les canaux interférents des autres utilisateurs. Cela conduit à l'utilisation de deux CNNs distincts au lieu du $\text{CNN}_{\mathbf{E}}$ unique utilisé sur la liaison montante. La Section 5.3.2 donne plus d'information sur l'utilisation de ces deux CNNs.

Démappage amélioré par DL

Une conséquence de l'estimation et de l'égalisation imparfaites est le vieillissement du canal, qui entraîne des distorsions résiduelles sur les signaux égalisés. Un démappeur traditionnel opère indépendamment sur chaque RE et ne voit donc qu'un seul symbole égalisé à la fois. En revanche, nous proposons d'utiliser un CNN, appelé CNN_{Dmp}, pour effectuer un démappage conjoint de l'ensemble de la RG. En traitant conjointement tous les symboles égalisés, ce CNN peut estimer et corriger les effets du vieillissement du canal pour calculer de meilleurs rapports de log-vraisemblance (log-likelihood ratios, LLRs). L'entrée de CNN_{Dmp} est de dimension $M \times N \times 6$ et contient les indices des symboles et des sous-porteuses pour chaque RE, le SNR, les parties réelles et imaginaires des symboles égalisés, et les variances du bruit du canal après égalisation. La sortie de CNN_{Dmp} est de dimension $M \times N \times Q$ et correspond aux $\text{LLRs}^{(k)}$ prédits sur la RG pour un utilisateur k. Comme avec un récepteur conventionnel, le démappage

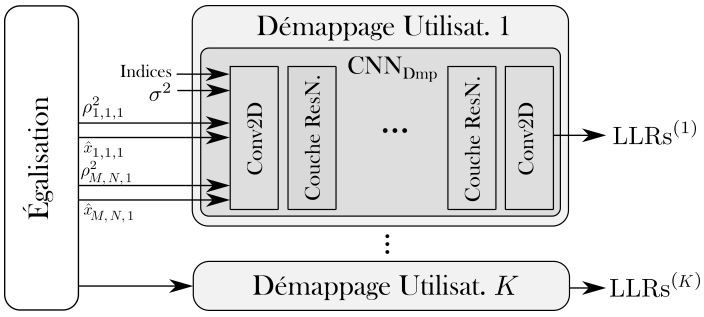


FIGURE B.18 : Démappage neuronal en liaison montante.

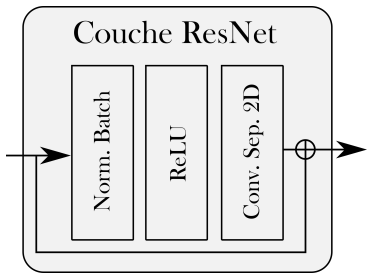


FIGURE B.19 : Illustration d'une couche ResNet.

est effectué indépendamment pour chaque utilisateur afin de rendre l'architecture facilement adaptable à un nombre variable d'utilisateurs. Le démappeur $\mathrm{CNN}_{\mathrm{Dmp}}$ est présenté dans la Figure B.18, qui décrit le processus de démappage de la liaison montante.

Architecture des CNNs

Tous les CNN présentés ci-dessus partagent les mêmes blocs : couches convolutionelles 2D, couches denses et couches ResNet personnalisées. Ces couches ResNet sont constituées d'une couche de normalisation par batch, d'une fonction d'activation ReLU, d'une couche convolutive séparable 2D et enfin de l'ajout de l'entrée, comme le montre la Figure B.19. Les convolutions séparables sont moins coûteuses en termes de calcul tout en maintenant des performances similaires à celles des couches convolutionnelles classiques [31]. Les détails des architectures de tous les CNNs sont donnés dans le Tableau 5.1.

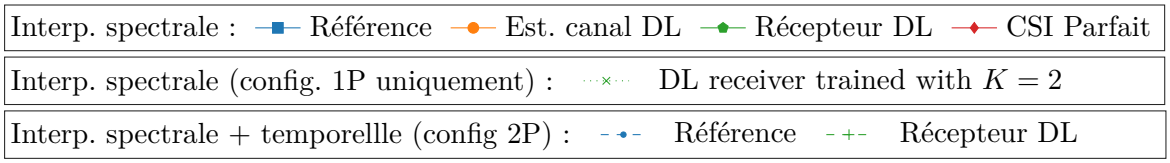
B.4.4 Évaluations

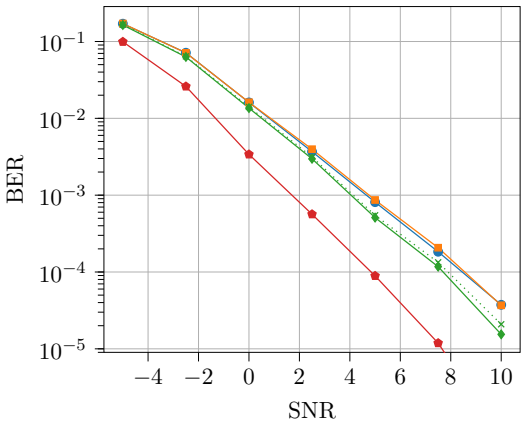
Dispositif d'entraînement et d'évaluation

Pour un entraînement et une évaluation réalistes, les réalisations de canal ont été générées avec QuaDRiGa version 2.0.0 [85]. Le nombre d'utilisateurs a été fixé à $K = 4$, sauf pour les liaisons descendantes à haute vitesse où il a été réduit à $K = 2$, et le nombre d'antennes de la BS a été fixé à $L = 16$. Les RGs étaient composées de $N = 72$ sous-porteuses, avec une fréquence centrale de 3.5 GHz et un espacement des sous-porteuses de 15 kHz. Une QAM utilisant un code de Gray a été utilisée avec $Q = 4$ bits par utilisation du canal sur la liaison montante et $Q = 2$ bits par utilisation du canal sur la liaison descendante. Les récepteurs ont été entraînés à des vitesses allant de 0 à 45 $\mathrm{km\,h^{-1}}$ et de 50 à 130 $\mathrm{km\,h^{-1}}$. Trois gammes de vitesses faibles ont été considérées pour les essais avec la configuration de pilotes 1P : 0 à 15 $\mathrm{km\,h^{-1}}$, 15 à 30 $\mathrm{km\,h^{-1}}$, et 30 à 45 $\mathrm{km\,h^{-1}}$. De même, trois gammes de vitesses élevées ont été considérées avec la configuration de pilotes 2P : 50 à 70 $\mathrm{km\,h^{-1}}$, 80 à 100 $\mathrm{km\,h^{-1}}$, et 110 à 130 $\mathrm{km\,h^{-1}}$. Les évaluations ont été effectuées séparément pour chaque plage de vitesse en utilisant un code LDPC standard IEEE 802.11n de longueur 1296 bits. Des taux de codage de $\frac{1}{2}$ et $\frac{1}{3}$ ont été utilisés respectivement pour les transmissions en liaison montante et en liaison descendante. Chaque entraînement a été effectué à l'aide de batchs de taille $B_S = 27$ de sorte que le nombre total de bits envoyés pour chaque utilisateur soit un multiple de la longueur du code.

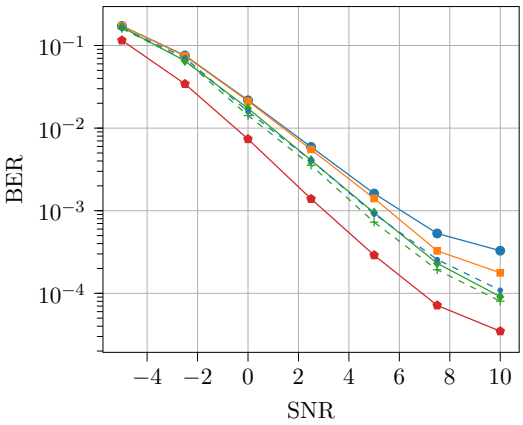
Résultats des simulations sur la liaison montante

Différents systèmes ont été comparés dans les simulations de liaison montante. Le premier est le système de référence pour la liaison montante présenté dans la Section B.4.2. Le deuxième, appelé 'estimateur de canal DL', utilise l'estimateur de canal amélioré par DL mais est entraîné et testé avec un démappeur standard. Le troisième est le récepteur amélioré par DL, qui utilise à la fois l'estimateur de canal amélioré et le démappeur amélioré. Nous l'appelons 'récepteur

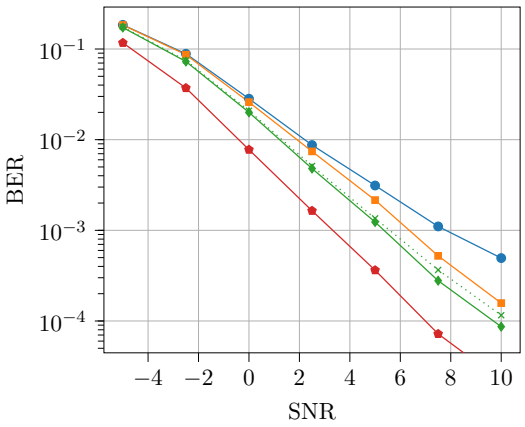




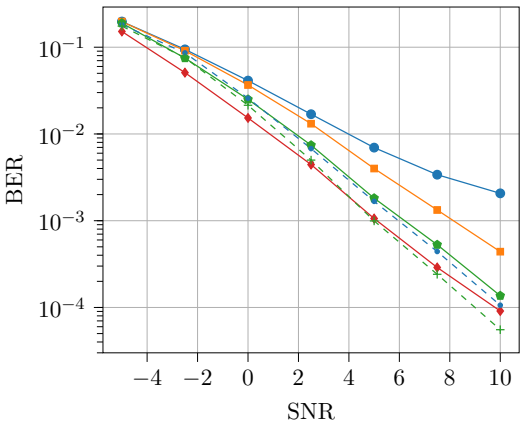
(a) Configuration 1P de 0 à 5 km h^{-1}.



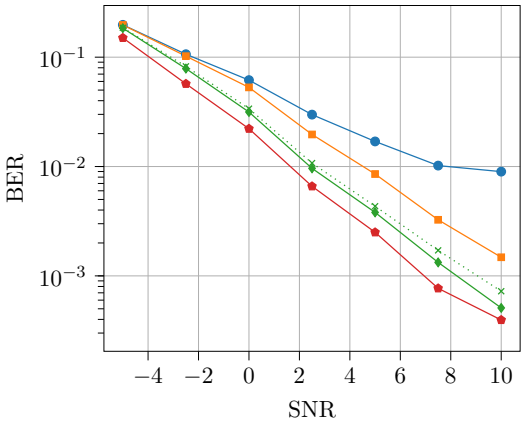
(b) Configuration 2P de 40 à 60 km h^{-1}.



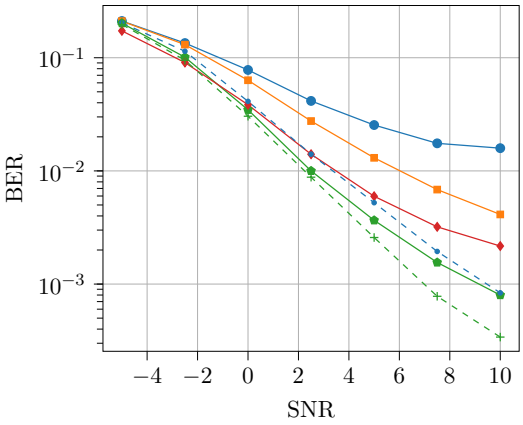
(c) Configuration 1P de 10 à 20 km h^{-1}.



(d) Configuration 2P de 70 à 90 km h^{-1}.



(e) Configuration 1P de 25 à 35 km h^{-1}.



(f) Configuration 2P de 110 à 130 km h^{-1}.

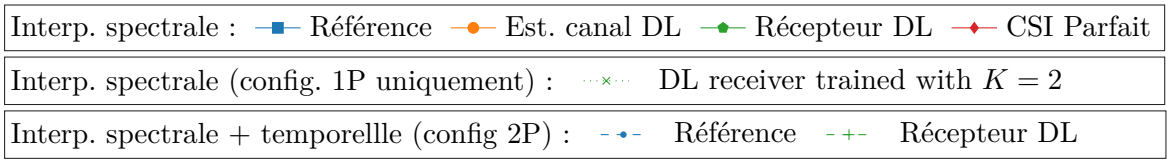
FIGURE B.20 : BERs en liaison montante obtenus avec les configurations de pilotes 1P et 2P.

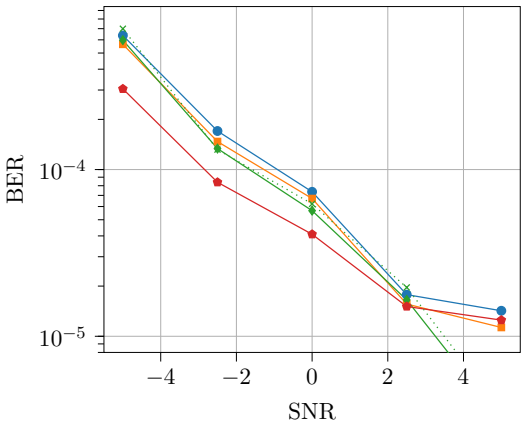
DL'. L'évaluation séparée de l'estimateur de canal DL nous permet de mieux comprendre le rôle des deux composants dans les gains observés. Une système de référence idéal avec une connaissance parfaite du canal au niveau des REs transportant les pilotes et de $\mathbf{E}$ est également considéré, et est désigné sous le nom de 'Parfaite connaissance des informations sur l'état du canal (channel state information, CSI)', ou 'CSI parfait'. Tous les systèmes ont utilisé une interpolation spectrale suivie d'une approximation NIRE pour l'estimation du canal. Des simulations supplémentaires ont été effectuées pour la configuration de pilotes 2P en utilisant l'interpolation spectrale et temporelle pour le système de référence et le récepteur DL. Enfin, un récepteur DL entraîné avec seulement $K = 2$ utilisateurs a également été évalué.

Les résultats de la simulation pour la configuration 1P sont présentés dans la première colonne de la Figure B.20 pour les trois plages de vitesse différentes. Aux vitesses comprises entre 0 et 15 $\mathrm{km\,h^{-1}}$, on constate que le récepteur DL n'apporte aucune amélioration. Dans la plage de vitesse la plus élevée (Figure B.20e), le récepteur DL permet des gains de 3 dB par rapport au système de référence pour un BER codé de 10^{-2}. On constate que le récepteur DL entraîné avec seulement deux utilisateurs est presque aussi performant que le récepteur entraîné avec quatre utilisateurs pour toutes les vitesses considérées, ce qui démontre l'adaptabilité du système proposé vis-à-vis du nombre d'utilisateurs. Les résultats pour la configuration 2P sont présentés à la deuxième colonne de la Figure B.20. Les gains fournis par le récepteur DL suivent la même tendance que pour la configuration 1P. Plus précisément, le récepteur DL est le seul système qui permet d'obtenir un BER codé de 10^{-3} pour les vitesses les plus élevées avec l'interpolation spectrale uniquement. En effet, à des vitesses élevées, le démappeur appris est encore capable d'atténuer les effets du vieillissement du canal, alors que même le système de référence avec CSI parfait souffre de signaux égalisés fortement perturbés. L'utilisation des interpolations spectrales et temporelles réduit les gains fournis par le récepteur DL, ce qui peut s'expliquer par les meilleures estimations du canal conduisant à un moindre vieillissement du canal, mais ils représentent toujours un écart de 2.2 dB à un BER codé de 10^{-3} pour les vitesses les plus élevées. Dans l'ensemble, on peut voir que seule la combinaison d'une estimation des statistiques d'erreur d'estimation du canal basée sur des CNNs et d'un démappage basé sur un CNN permet d'obtenir des gains pour les deux configurations de pilotes et pour toutes les intervalles de vitesse.

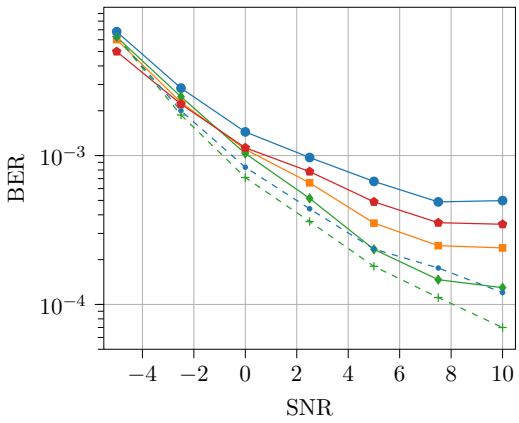
Résultats des simulations sur la liaison descendante

Quatre systèmes ont été évalués sur la liaison descendante pour les intervalles de vitesse considérées. Le premier est le système de référence présenté dans la Section B.4.2. Dans le second, appelé 'estimateur de canal DL', chaque utilisateur utilise l'estimateur de canal amélioré par DL et est entraîné et testé avec un démappeur standard. Le troisième est le récepteur amélioré par DL qui utilise à la fois l'estimateur de canal amélioré et le démappeur DL pour chaque utilisateur, et est appelé 'récepteur DL'. Le dernier schéma a un CSI parfait, c'est-à-dire que tous les utilisateurs connaissent parfaitement à la fois le canal aux REs portant des pilotes et les variances des erreurs d'estimation partout. Comme pour la liaison

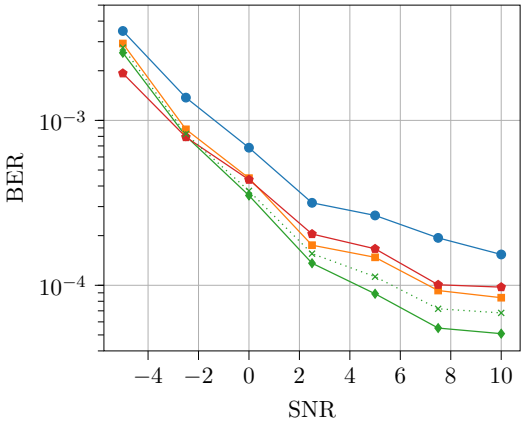




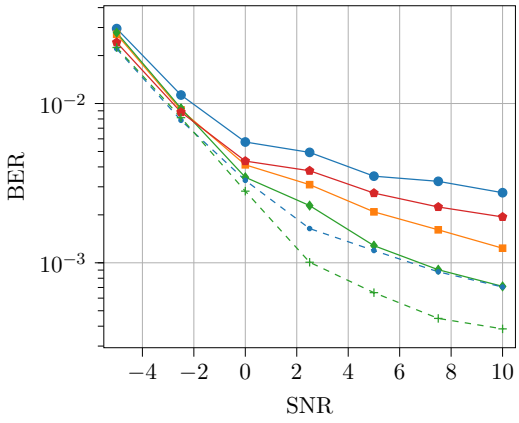
(a) Configuration 1P de 0 à 5 km h^{-1}.



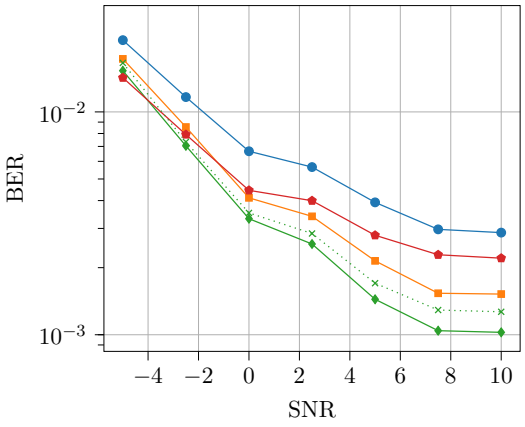
(b) Configuration 2P de 40 à 60 km h^{-1}.



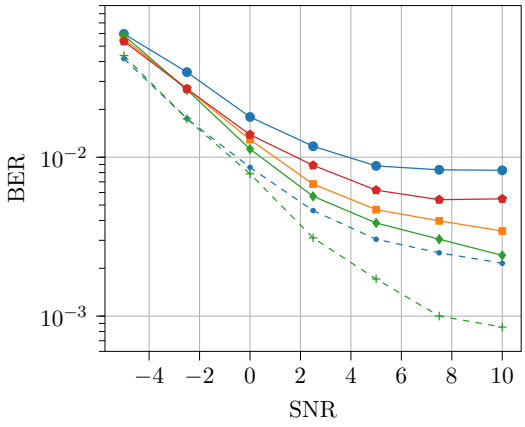
(c) Configuration 1P de 10 à 20 km h^{-1}.



(d) Configuration 2P de 70 à 90 km h^{-1}.



(e) Configuration 1P de 25 à 35 km h^{-1}.



(f) Configuration 2P de 110 à 130 km h^{-1}.

FIGURE B.21 : BERs en liaison descendante obtenus avec les config. de pilotes 1P et 2P.

montante, tous les systèmes ont utilisé l'interpolation spectrale avec approximation NIRE pour l'estimation du canal, mais des simulations supplémentaires ont été effectuées pour la configuration de pilotes 2P en utilisant l'interpolation spectrale et temporelle.

Les évaluations de la configuration de pilotes 1P en liaison descendante sont présentées dans la première colonne de la Figure B.21. Comme prévu, les gains permis par l'estimateur de canal et le démappeur appris augmentent avec la vitesse. On peut également observer que le récepteur DL entraîné avec seulement $K = 2$ utilisateurs est capable de s'approcher des performances du récepteur DL entraîné avec quatre utilisateurs. Les évaluations de la configuration de pilotes 2P sont présentées sur la deuxième colonne de la Figure B.21 pour des vitesses plus élevées. En utilisant les interpolations spectrales et temporelles, le récepteur DL offre toujours des gains significatifs à des vitesses élevées, étant le seul à atteindre un BER de 10^{-3} dans l'intervalle de vitesse de 110 à 130 $\mathrm{km\,h^{-1}}$. Dans tous les scénarios de liaison descendante, à l'exception de l'intervalle de vitesse le plus lent, l'estimateur de canal DL est plus performant que le système de référence avec un CSI parfait. Nous supposons que cela est dû au fait que le système de référence suppose que le bruit de la liaison descendante après égalisation est distribué de manière gaussienne et est non corrélé au signal transmis, ce qui est généralement faux.

B.4.5 Conclusion

Le présent chapitre propose une nouvelle stratégie hybride pour la conception de récepteurs MU-MIMO améliorés par des composants DL. Notre architecture s'appuie sur un récepteur MU-MIMO traditionnel et l'améliore avec des composants DL qui sont entraînés pour maximiser les performances de bout en bout du système. Cette approche ne nécessite aucune connaissance du canal pour l'entraînement, elle est interprétable et facilement adaptable à un nombre quelconque d'utilisateurs. Des évaluations en liaison montante et descendante ont été réalisées avec plusieurs intervalles de vitesses et deux configurations de pilotes différentes sur des modèles de canal conformes à la norme 3GPP. Les résultats révèlent que l'architecture proposée exploite efficacement la structure OFDM pour obtenir des gains tangibles à faible vitesse et des gains importants à vitesse élevée par rapport à un récepteur traditionnel. Nous pensons que de telles architectures, permettant d'améliorer les performances tout en restant conformes aux standards et raisonnablement complexes, pourraient être déployées dans les BS de la prochaine génération de systèmes de communication sans fil. Cependant, il a été démontré que des gains supplémentaires peuvent encore être obtenus avec un récepteur entièrement basé sur un NN qui traitent conjointement tous les utilisateurs [82], contrairement à notre approche dans laquelle ils sont traités indépendamment. Débloquer ces gains tout en préservant l'adaptabilité et l'interprétabilité des architectures conventionnelles reste un défi important qui n'a pas encore été résolu.

B.5 Réflexions Finales

Au cours de ce manuscrit, nous avons étudié trois stratégies qui visent à apporter les avantages du DL à la couche physique des systèmes de communication. La première stratégie, que nous avons appelée l'optimisation par blocs basée sur des NN, a été utilisée pour concevoir un NN qui pourrait remplacer les détecteurs MU-MIMO actuels. Mais ces détecteurs basés sur un NN supposent souvent un CSI parfait et sont toujours entraînés à optimiser leurs propres performances uniquement, ce qui ne garantit pas l'optimalité au niveau du système en entier. La deuxième stratégie, dans laquelle le système est optimisé de bout en bout en implémentant les émetteurs-récepteurs en tant que NNs, a été discutée dans la Section B.3. Elle permet une optimisation plus poussée du traitement du signal en émission et en réception puisqu'elle ne doit pas se conformer aux protocoles existants. Bien que prometteurs, l'émetteur et le récepteur correspondants sont trop complexes pour une implémentation à court terme, ne répondent pas aux standards actuels et sont limités aux transmissions SISO. Enfin, une troisième stratégie hybride a été proposée dans la Section B.4. Elle consiste à insérer des composants basés sur du DL dans une architecture traditionnelle qui est entraînée de bout en bout. Nous avons utilisé cette stratégie pour améliorer un récepteur MO-MIMO classique avec plusieurs CNNs. Les résultats indiquent que notre récepteur réalise des gains dans tous les scénarios, tant en liaison montante qu'en liaison descendante, et surtout à des vitesses élevées. Plus important encore, il préserve l'adaptabilité des architectures conventionnelles à un nombre variable d'utilisateurs et pourrait devenir conforme aux standards. Globalement, cette stratégie semble être la plus adaptée à une utilisation dans un avenir proche.

Le déploiement du DL dans la couche physique se fera probablement en plusieurs phases. La première correspond à l'intégration de blocs basés sur du DL dans les récepteurs traditionnels de la BS, entraînés selon la stratégie basée sur les blocs ou la stratégie hybride. Comme le DL est avant tout un problème de données, les gains apportés par l'intégration du DL sont liés à notre capacité à constituer des ensembles de données suffisants. La collecte de données représentatives d'environnements divers et variés est donc essentielle pour garantir la fiabilité de ces composants. Ces défis sont exacerbés sur les appareils mobiles, dans lesquels l'intégration du DL pourrait correspondre à une deuxième phase de déploiement. Les gains devraient s'en trouver améliorés puisque les modèles DL pourront prendre en compte les distorsions et les spécificités des canaux qui ne sont pas présentes dans les modèles traditionnels. Cette flexibilité est particulièrement intéressante pour les futurs réseaux 6G, qui devraient prendre en charge un large éventail de cas d'utilisation, comme les communications entre véhicules ou les sous-réseaux de petite échelle. La disponibilité croissante d'accélérateurs DL aux deux extrémités de la communication devrait permettre à terme l'émergence d'émetteurs-récepteurs entièrement basés sur des NNs, entraînés en utilisant la stratégie de bout en bout. Cependant, l'apprentissage conjoint d'émetteurs et de récepteurs conçus par différents fabricants représente un défi majeur pour l'industrie des télécommunications. Les futurs standards devront donc passer d'une réglementation du comportement des algorithmes de communication à une optimisation de bout en bout des systèmes basés sur des NNs. Les prochaines phases du

déploiement du DL dans les systèmes de communication pourraient finalement se concentrer sur l'apprentissage des couches physique et de contrôle d'accès au support (medium access control, MAC), donnant finalement naissance aux premières radios entièrement concues et controllées par intelligence artificielle.

Acronyms, Symbols, Figures and Tables

List of acronyms

3GPP 3rd Generation Partnership Project

ACLR adjacent channel leakage ratio

AI artificial intelligence

AMP approximate message passing

AWGN additive white Gaussian noise

BCE binary cross-entropy

BER bit error rate

BLER block error rate

BMD bit-metric decoding

BS base station

CCDF complementary cumulative distribution function

CE cross-entropy

CNN convolutional neural network

CP cyclic prefix

CSI channel state information

DFT discrete Fourier transform

DL deep learning

ELU exponential linear unit

FBS frequency baseband symbol

i.i.d. independent and identically distributed

IDFT inverse discrete Fourier transform

ISI intersymbol interference

LDPC low-density parity-check

LLR log-likelihood ratio

LMMSE linear minimum mean squared error

MAC medium access control

MAP maximum a posteriori

MIMO multiple-input multiple-output

ML machine learning

MSE mean squared error

MU multi-user

MU-MIMO multi-user multiple-input multiple-output

NIRE nearest interpolated resource element

NLOS non-line of sight

NN neural network

NR new radio

OFDM orthogonal frequency-division multiplexing

PA power amplifier

PAPR peak-to-average power ratio

PRT peak reduction tone

PSD power spectral density

QAM quadrature amplitude modulation

QPSK quadrature phase-shift keying

RE resource element

ReLU rectified linear unit

RG resource grid

RNN recurrent neural network

SER symbol error rate

SGD stochastic gradient descent

SIMO single-input multiple-output

SISO single-input single-output

SNR signal-to-noise ratio

SU single-user

SVD singular value decomposition

TDD time-division duplexing

TR tone reservation

ULA uniform linear array

UMi urban microcell

w.r.t. with respect to

List of Symbols

$\mathbf{A}^{(i)}$	Matrix used in i^{th} iteration of an iterative decoder(Chapter 3)
$\mathbf{B}$	Tensor of bits
B_S	Batch size
$\mathbf{C}$	Downlink normalization precoding downlink matrix (Chapter 5)
$\mathcal{C}$	Constellation
C	Achievable rate
$\mathbf{d}_m, d_{m,n\in\mathcal{D}}$	(Vector of) FBSs carrying data signals (Chapter 4)
$\mathbf{D}$	Uplink post-equalization rescaling matrix (Chapter 5)
$\mathcal{D}, D$	Set and number of subcarriers carrying data (Chapter 4)
$\mathbf{E}$	Channel estimation error spatial covariance (Chapter 5)
E_{A_m}, E_{I_m}	In-band and total energies of an OFDM symbol
$\mathbf{F}^{\mathsf{H}}$	Inverse Fourier Transform
$\mathbf{G}$	Equivalent downlink channel (Chapter 5)
$\mathbf{H}$	Tensor/matrix of channel coefficients
$\mathbf{J}$	Matrix for the in-band energy of an OFDM symbol (Chapter 4)
$\mathbf{K}$	Matrix for the total energy of an OFDM symbol (Chapter 4)
K	Number of users in MU-MIMO systems
k	User index
$\mathbf{L}$	Number of antennas at the BS
l	BS antenna index
$\mathcal{M}, M$	Set and number of OFDM symbols
m	OFDM symbol index
$\mathbf{N}$	Noise tensor/matrix
$\mathcal{N}, N$	Subset and number of subcarriers
n	Subcarrier index
$\lvert\mathcal{P}_M\rvert$, $\lvert\mathcal{P}_N\rvert$	Number of pilots in the time and frequency domain
$\widehat{P}(\cdot), P(\cdot)$	(Predicted) probability
$\mathcal{P}^{(k)}$	Pilot pattern for the k^{th} user (Chapter 5)
$\mathbf{p}$	Pilots
Q	Number of bits per channel uses
q	Bit index
$\mathbf{q}_{m,n}$	Downlink noise vector (Chapter 5)
$\mathbf{Q}$, $\mathbf{Q_A}$, $\mathbf{Q_B}$	QR-decomposition
$\mathbf{r}_m, r_{m,n\in\mathcal{R}}$	(Vector of) FBS carrying peak-reduction signal

$\mathbf{R}$, $\mathbf{R_A}$	QR-decomposition
$\mathcal{R}, R$	Set of number of subcarriers carrying peak-reduction signals (Chapter 4)
$\mathbf{S}$	Tensor of unprecoded downlink FBSs (Chapter 5)
$S_m(f)$	Baseband spectrum
$s_m(t)$	Time-domain signal
$\mathbf{T}$	Tensor of precoded downlink symbols (Chapter 5)
$T, T^{\mathrm{CP}}, T^{\mathrm{tot}}$	Duration of an OFDM symbol, of its CP, and total duration
$\mathbf{U}$	Downlink received signal (Chapter 5)
$\mathbf{V}^{(k)}$	Equivalent downlink channel estimation error variances (Chapter 5)
$\mathbf{W}$	LMMSE matrix
$\hat{x}$	Estimated symbols after equalization
$\hat{\hat{x}}$	Hard decision on the estimated symbols ($\hat{\hat{x}} \in \mathcal{C}$)
$\mathbf{z}$	Discrete time-domain signal (Chapter 4)
$\boldsymbol{\theta}$, $\boldsymbol{\psi}$	Set of trainable parameters
$\nu(t)$	Ration between the instantaneous and average power of a signal (Chapter 4)
α, β, γ	Parameters of the exponential decay model (Chapter 5)
σ^2	General notation for a noise variance
τ^2	Variance of the post-equalization downlink noise (Chapter 5)
ϵ	PAPR threshold
η	Learning rate (Chapter 2) or code rate (Chapter 3, 4, and 5)
ρ^2	Variance of the post-equalization uplink noise (Chapter 5)
$\xi_{m,n,k}$	Post-equalization downlink noise (Chapter 5)
$\zeta_{m,n,k}$	Post-equalization downlink noise (Chapter 5)
$\boldsymbol{\Theta}$	Matrix to be optimized in the iterative detection algorithm (Chapter 3)
$\boldsymbol{\Sigma}$	General notation for a channel covariance matrix
$\boldsymbol{\Omega}$	Covariance matrix for the main downlink channel (Chapter 5)
$\boldsymbol{\Psi}$	Covariance matrix for the interfering downlink channels (Chapter 5)

List of Figures

List of Tables

Bibliography

[1] W. S. McCulloch and W. Pitts, "A Logical Calculus of the Ideas Immanent in Nervous Activity", *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[2] James Bennett and Stan Lanning and Netflix Netflix, "The Netflix Prize", in *In KDD Cup and Workshop in conjunction with KDD*, 2007.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., vol. 25, Curran Associates, Inc., 2012.

[4] C. Weng, D. Yu, S. Watanabe, and B.-H. F. Juang, "Recurrent Deep Neural Networks for Robust Speech Recognition", in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 5532–5536. DOI: 10.1109/ICASSP.2014.6854661.

[5] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2Face: Real-Time Face Capture and Reenactment of RGB Videos", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2387–2395.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423.

[7] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of Go with deep neural networks and tree search", *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database", in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[9] T. S. Rappaport *et al.*, *Wireless Communications: Principles and Practice*. Prentice Hall PTR New Jersey, 1996, vol. 2.

[10] E. Zehavi, "8-PSK Trellis Codes for a Rayleigh Channel", *IEEE Transactions on Communications*, vol. 40, no. 5, pp. 873–884, 1992. DOI: 10.1109/26.141453.

[11] D. Chen, B. Sheu, and T. Berger, "A Compact Neural Network Based CDMA Receiver for Multimedia Wireless Communication", in *Proceedings International Conference on Computer Design. VLSI in Computers and Processors*, 1996, pp. 99–103. DOI: 10.1109/ICCD.1996.563540.

[12] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning", in *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2016, pp. 341–346. DOI: 10.1109/ALLERTON.2016.7852251.

[13] T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer", *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017. DOI: 10.1109/TCCN.2017.2758370.

[14] S. Dörner, S. Cammerer, J. Hoydis, and S. t. Brink, "Deep Learning Based Communication Over the Air", *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 132–143, 2018. DOI: 10.1109/JSTSP.2017.2784180.

[15] F. A. Aoudia and J. Hoydis, "Model-Free Training of End-to-End Communication Systems", *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 11, pp. 2503–2516, 2019. DOI: 10.1109/JSAC.2019.2933891.

[16] F. A. Aoudia and J. Hoydis, "End-to-end Learning for OFDM: From Neural Receivers to Pilotless Communication", *IEEE Transactions on Wireless Communications*, pp. 1–1, 2021. DOI: 10.1109/TWC.2021.3101364.

[17] A. Valcarce and J. Hoydis, "Towards Joint Learning of Optimal MAC Signaling and Wireless Channel Access", *IEEE Transactions on Cognitive Communications and Networking*, pp. 1–1, 2021. DOI: 10.1109/TCCN.2021.3080677.

[18] J. Hoydis, F. A. Aoudia, A. Valcarce, and H. Viswanathan, "Toward a 6G AI-Native Air Interface", *IEEE Communications Magazine*, vol. 59, no. 5, pp. 76–81, May 2021, ISSN: 1558-1896. DOI: 10.1109/mcom.001.2001187.

[19] D. Gündüz, P. de Kerret, N. D. Sidiropoulos, D. Gesbert, C. R. Murthy, and M. van der Schaar, "Machine Learning in the Air", *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2184–2199, 2019. DOI: 10.1109/JSAC.2019.2933969.

[20] F. Alberge, "Deep Learning Constellation Design for the AWGN Channel With Additive Radar Interference", *IEEE Transactions on Communications*, vol. 67, no. 2, pp. 1413–1423, 2019. DOI: 10.1109/TCOMM.2018.2875721.

[21] S. Cammerer, F. A. Aoudia, S. Dörner, M. Stark, J. Hoydis, and S. ten Brink, "Trainable communication systems: concepts and prototype", *IEEE Transactions on Communications*, vol. 68, no. 9, pp. 5489–5503, 2020. DOI: 10.1109/TCOMM.2020.3002915.

[22] J. Downey, B. Hilburn, T. O'Shea, and N. West, "Machine Learning Remakes Radio", *IEEE Spectrum*, vol. 57, no. 5, pp. 35–39, 2020. DOI: 10.1109/MSPEC.2020.9078454.

[23] N. Kato, B. Mao, F. Tang, Y. Kawamoto, and J. Liu, "Ten Challenges in Advancing Machine Learning Technologies toward 6G", *IEEE Wireless Communications*, vol. 27, no. 3, pp. 96–103, 2020. DOI: 10.1109/MWC.001.1900476.

[24] S. Ali, W. Saad, N. Rajatheva, K. Chang, D. Steinbach, B. Sliwa, C. Wietfeld, K. Mei, H. Shiri, H.-J. Zepernick, T. M. C. Chu, I. Ahmad, J. Huusko, J. Suutala, S. Bhadauria, V. Bhatia, R. Mitra, S. Amuru, R. Abbas, B. Shao, M. Capobianco, G. Yu, M. Claes, T. Karvonen, M. Chen, M. Girnyk, and H. Malik, "6G White Paper on Machine Learning in Wireless Communication Networks", 2020. arXiv: 2004.13875 [cs.IT].

[25] H. Viswanathan and P. E. Mogensen, "Communications in the 6G Era", *IEEE Access*, vol. 8, pp. 57 063–57 074, 2020. DOI: 10.1109/ACCESS.2020.2981745.

[26] E. Björnson, J. Hoydis, and L. Sanguinetti, "Massive MIMO Networks: Spectral, Energy, and Hardware Efficiency", *Found. and Trends® in Signal Proc.*, vol. 11, no. 3-4, pp. 154–655, 2017, ISSN: 1932-8346. DOI: 10.1561/2000000093. [Online]. Available: http://dx.doi.org/10.1561/2000000093.

[27] S. Yang and L. Hanzo, "Fifty Years of MIMO Detection: The Road to Large-Scale MIMOs", *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 1941–1988, 2015. DOI: 10.1109/COMST.2015.2475242.

[28] O. Edfors, M. Sandell, J.-J. van de Beek, D. Landström, and F. Sjöberg, "An Introduction to Orthogonal Frequency-Division Multiplexing", 1996.

[29] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[30] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", 2017. arXiv: 1412.6980 [cs.LG].

[31] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", 2017. arXiv: 1704.04861 [cs.CV].

[32] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)", 2016. arXiv: 1511.07289 [cs.LG].

[33] M. Goutay, F. Ait Aoudia, and J. Hoydis, "Deep Reinforcement Learning Autoencoder with Noisy Feedback", in *2019 International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)*, 2019, pp. 1–6. DOI: 10.23919/WiOPT47501.2019.9144089.

[34] H. Ye, G. Y. Li, B.-H. F. Juang, and K. Sivanesan, "Channel Agnostic End-to-End Learning Based Communication Systems with Conditional GAN", in *2018 IEEE Globecom Workshops (GC Wkshps)*, 2018, pp. 1–5. DOI: 10.1109/GLOCOMW.2018.8644250.

[35] T. J. O'Shea, T. Roy, and N. West, "Approximating the Void: Learning Stochastic Channel Models from Observation with Variational Generative Adversarial Networks", in *2019 International Conference on Computing, Networking and Communications (ICNC)*, 2019, pp. 681–686. DOI: 10.1109/ICCNC.2019.8685573.

[36] G. Böcherer, "Achievable Rates for Probabilistic Shaping", 2018. arXiv: 1707.01134 [cs.IT].

[37] A. D. Pia, S. S. Dey, and M. Molinaro, "Mixed-Integer Quadratic Programming is in NP", *Mathematical Programming*, vol. 162, no. 1–2, pp. 225–240, Mar. 2017, ISSN: 0025-5610.

[38] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge university press, 2005.

[39] C. Jeon, R. Ghods, A. Maleki, and C. Studer, "Optimality of large MIMO detection via approximate message passing", in *2015 IEEE International Symposium on Information Theory (ISIT)*, 2015, pp. 1227–1231. DOI: 10.1109/ISIT.2015.7282651.

[40] J. Ma and L. Ping, "Orthogonal AMP", *IEEE Access*, vol. 5, pp. 2020–2033, 2017.

[41] S. Chaudhari, H. Kwon, and K.-B. Song, "Reliable and Low-Complexity MIMO Detector Selection using Neural Network", in *2020 International Conference on Computing, Networking and Communications (ICNC)*, 2020, pp. 608–613. DOI: 10.1109/ICNC47757.2020.9049677.

[42] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection", in *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2017, pp. 1–5. DOI: 10.1109/SPAWC.2017.8227772.

[43] A. Mohammad, C. Masouros, and Y. Andreopoulos, "Complexity-Scalable Neural-Network-Based MIMO Detection With Learnable Weight Scaling", *IEEE Transactions on Communications*, vol. 68, no. 10, pp. 6101–6113, 2020. DOI: 10.1109/TCOMM.2020.3007622.

[44] A. Balatsoukas-Stimming and C. Studer, "Deep Unfolding for Communications Systems: A Survey and Some New Directions", in *2019 IEEE International Workshop on Signal Processing Systems (SiPS)*, 2019, pp. 266–271. DOI: 10.1109/SiPS47522.2019.9020494.

[45] N. Samuel, T. Diskin, and A. Wiesel, "Learning to Detect", *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2554–2564, 2019. DOI: 10.1109/TSP.2019.2899805.

[46] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "A Model-Driven Deep Learning Network for MIMO Detection", *Proc. IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 584–588, 2018.

[47] M. Khani, M. Alizadeh, J. Hoydis, and P. Fleming, "Adaptive Neural Signal Detection for Massive MIMO", *IEEE Transactions on Wireless Communications*, vol. 19, no. 8, pp. 5635–5648, 2020. DOI: 10.1109/TWC.2020.2996144.

[48] L. Bertinetto, J. F. Henriques, J. Valmadre, P. Torr, and A. Vedaldi, "Learning feed-forward one-shot learners", in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29, Curran Associates, Inc., 2016.

[49] E. Zakharov, A. Shysheya, E. Burkov, and V. Lempitsky, "Few-Shot Adversarial Learning of Realistic Neural Talking Head Models", in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9458–9467. DOI: 10.1109/ICCV.2019.00955.

[50] M. Goutay, F. Ait Aoudia, and J. Hoydis, "Deep HyperNetwork-Based MIMO Detection", in *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2020, pp. 1–5. DOI: 10.1109/SPAWC48557.2020.9154283.

[51] N. Zilberstein, C. Dick, R. Doost-Mohammady, A. Sabharwal, and S. Segarra, "Robust MIMO Detection using Hypernetworks with Learned Regularizers", 2021. arXiv: 2110.07053 [eess.SP].

[52] K. Pratik, B. D. Rao, and M. Welling, "RE-MIMO: Recurrent and Permutation Equivariant Neural MIMO Detection", *IEEE Transactions on Signal Processing*, vol. 69, pp. 459–473, 2021. DOI: 10.1109/TSP.2020.3045199.

[53] D. Neumann, T. Wiese, and W. Utschick, "Learning the MMSE Channel Estimator", *IEEE Transactions on Signal Processing*, vol. 66, no. 11, pp. 2905–2917, 2018. DOI: 10.1109/TSP.2018.2799164.

[54] P. Dong, H. Zhang, G. Y. Li, I. S. Gaspar, and N. NaderiAlizadeh, "Deep CNN-Based Channel Estimation for mmWave Massive MIMO Systems", *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 5, pp. 989–1000, 2019. DOI: 10.1109/JSTSP.2019.2925975.

[55] M. B. Mashhadi and D. Gündüz, "Pruning the Pilots: Deep Learning-Based Pilot Design and Channel Estimation for MIMO-OFDM Systems", *IEEE Transactions on Wireless Communications*, vol. 20, no. 10, pp. 6315–6328, 2021. DOI: 10.1109/TWC.2021.3073309.

[56] O. Shental and J. Hoydis, ""Machine LLRning": Learning to Softly Demodulate", in *2019 IEEE Globecom Workshops (GC Wkshps)*, 2019, pp. 1–7. DOI: 10.1109/GCWkshps45667.2019.9024433.

[57] A. Öztekin and E. Erçelebi, "An Efficient Soft Demapper for APSK Signals Using Extreme Learning Machine", *Neural Computing and Applications*, vol. 31, no. 10, pp. 5715–5727, 2019.

[58] M. Schädler, G. Böcherer, and S. Pachnicke, "Soft-Demapping for Short Reach Optical Communication: A Comparison of Deep Neural Networks and Volterra Series", *Journal of Lightwave Technology*, vol. 39, no. 10, pp. 3095–3105, 2021. DOI: 10.1109/JLT.2021.3056869.

[59] D. Ha, A. Dai, and Q. V. Le, "HyperNetworks", 2016. arXiv: 1609.09106 [cs.LG].

[60] Y. S. Nasir and D. Guo, "Multi-Agent Deep Reinforcement Learning for Dynamic Power Allocation in Wireless Networks", *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2239–2250, 2019. DOI: 10.1109/JSAC.2019.2933973.

[61] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is All you Need", in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc., 2017.

[62] P. Putzky and M. Welling, "Recurrent Inference Machines for Solving Inverse Problems", 2017. arXiv: 1706.04008 [cs.NE].

[63] J. Hoydis, F. Ait Aoudia, A. Valcarce, and H. Viswanathan, "Toward a 6G AI-Native Air Interface", 2021. arXiv: 2012.08285 [cs.NI].

[64] M. Goutay, F. Ait Aoudia, J. Hoydis, and J.-M. Gorce, "Learning OFDM Waveforms with PAPR and ACLR Constraints", 2021. arXiv: 2110.10987 [cs.IT].

[65] Y.-C. Wang and Z.-Q. Luo, "Optimized Iterative Clipping and Filtering for PAPR Reduction of OFDM Signals", *IEEE Transactions on Communications*, vol. 59, no. 1, pp. 33–37, 2011. DOI: 10.1109/TCOMM.2010.102910.090040.

[66] B. Krongold and D. Jones, "PAR Reduction in OFDM via Active Constellation Extension", *IEEE Transactions on Broadcasting*, vol. 49, no. 3, pp. 258–268, 2003. DOI: 10.1109/TBC.2003.817088.

[67] B. Krongold and D. Jones, "An Active-Set Approach for OFDM PAR Reduction via Tone Reservation", *IEEE Transactions on Signal Processing*, vol. 52, no. 2, pp. 495–509, 2004. DOI: 10.1109/TSP.2003.821110.

[68] M. Zhang, M. Liu, and Z. Zhong, "Neural Network Assisted Active Constellation Extension for PAPR Reduction of OFDM System", in *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*, 2019, pp. 1–6. DOI: 10.1109/WCSP.2019.8928056.

[69] B. Wang, Q. Si, and M. Jin, "A Novel Tone Reservation Scheme Based on Deep Learning for PAPR Reduction in OFDM Systems", *IEEE Communications Letters*, vol. 24, no. 6, pp. 1271–1274, 2020. DOI: 10.1109/LCOMM.2020.2980832.

[70] X. Wang, N. Jin, and J. Wei, "A Model-Driven DL Algorithm for PAPR Reduction in OFDM System", *IEEE Communications Letters*, vol. 25, no. 7, pp. 2270–2274, 2021. DOI: 10.1109/LCOMM.2021.3076605.

[71] L. Li, C. Tellambura, and X. Tang, "Improved Tone Reservation Method Based on Deep Learning for PAPR Reduction in OFDM System", in *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*, 2019, pp. 1–6. DOI: 10.1109/WCSP.2019.8928103.

[72] M. Kim, W. Lee, and D.-H. Cho, "A Novel PAPR Reduction Scheme for OFDM System Based on Deep Learning", *IEEE Communications Letters*, vol. 22, no. 3, pp. 510–513, 2018. DOI: 10.1109/LCOMM.2017.2787646.

[73] C. Tellambura, "Computation of the Continuous-Time PAR of an OFDM Signal with BPSK Subcarriers", *IEEE Communications Letters*, vol. 5, no. 5, pp. 185–187, 2001. DOI: 10.1109/4234.922754.

[74] T. V. Luong, Y. Ko, M. Matthaiou, N. A. Vien, M.-T. Le, and V.-D. Ngo, "Deep Learning-Aided Multicarrier Systems", *IEEE Transactions on Wireless Communications*, vol. 20, no. 3, pp. 2109–2119, 2021. DOI: 10.1109/TWC.2020.3039180.

[75] B. Karanov, M. Chagnon, F. Thouin, T. A. Eriksson, H. Bülow, D. Lavery, P. Bayvel, and L. Schmalen, "End-to-End Deep Learning of Optical Fiber Communications", *Journal of Lightwave Technology*, vol. 36, no. 20, pp. 4843–4855, 2018. DOI: 10.1109/JLT.2018.2865109.

[76] F. A. Aoudia and J. Hoydis, *Waveform Learning for Next-Generation Wireless Communication Systems*, 2021. arXiv: 2109.00998 [cs.IT].

[77] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization", *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[78] N. Andgart, "Peak and Power Reduction in Multicarrier Communication Systems", eng, Ph.D. dissertation, Lund University, 2005, ISBN: 91-7167-035-1. [Online]. Available: https://lup.lub.lu.se/search/ws/files/5589789/545619.pdf.

[79] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. Academic press, 2014.

[80] J. Nocedal and S. Wright, *Numerical Optimization*. Springer Science & Business Media, 2006.

[81] M. Honkala, D. Korpi, and J. M. J. Huttunen, "DeepRx: Fully Convolutional Deep Learning Receiver", *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 3925–3940, 2021. DOI: 10.1109/TWC.2021.3054520.

[82] D. Korpi, M. Honkala, J. M. Huttunen, and V. Starck, "DeepRx MIMO: Convolutional MIMO Detection with Learned Multiplicative Transformations", in *ICC 2021 - IEEE International Conference on Communications*, 2021, pp. 1–7. DOI: 10.1109/ICC42927.2021.9500518.

[83] M. Goutay, F. Ait Aoudia, J. Hoydis, and J.-M. Gorce, "Machine Learning for MU-MIMO Receive Processing in OFDM Systems", *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2318–2332, 2021. DOI: 10.1109/JSAC.2021.3087224.

[84] K. He, X. Zhang, S. Ren, and J. Sun, "Identity Mappings in Deep Residual Networks", in *European Conference on Computer Vision (ECCV)*, Springer, 2016, pp. 630–645.

[85] S. Jaeckel, L. Raschkowski, K. Börner, and L. Thiele, "QuaDRiGa: A 3-D Multi-Cell Channel Model With Time Evolution for Enabling Virtual Field Trials", *IEEE Transactions on Antennas and Propagation*, vol. 62, no. 6, 2014. DOI: 10.1109/TAP.2014.2310220.

[86] F. A. Eras, I. A. Carreño, T. Borja, D. J. Reinoso, L. Urquiza-Aguiar, and M. C. P. Paredes, "Analysis of the PAPR Behavior of the OFDM Passband Signal", 2019.

[87] H. Ye, G. Y. Li, and B. Juang, "Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems", *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, 2018. DOI: 10.1109/LWC.2017.2757490.

[88] Z. Zhao, M. C. Vuran, F. Guo, and S. D. Scott, "Deep-Waveform: A Learned OFDM Receiver Based on Deep Complex-valued Convolutional Networks", 2021. arXiv: 1810.07181 [eess.SP].

[89] Y. Zhang, A. Doshi, R. Liston, W.-T. Tan, X. Zhu, J. G. Andrews, and R. W. Heath, "DeepWiPHY: Deep Learning-Based Receiver Design and Dataset for IEEE 802.11ax Systems", *IEEE Transactions on Wireless Communications*, vol. 20, no. 3, pp. 1596–1611, 2021. DOI: 10.1109/TWC.2020.3034610.

[90] Z. Zhou, L. Liu, and H.-H. Chang, "Learning for Detection: MIMO-OFDM Symbol Detection Through Downlink Pilots", *IEEE Transactions on Wireless Communications*, vol. 19, no. 6, pp. 3712–3726, 2020. DOI: 10.1109/TWC.2020.2976004.

[91] 3rd Generation Partnership Project; Technical Specification Group Radio Access Network, "Evolved Universal Terrestrial Radio Access (E-UTRA); Base Station (BS) Conformance Testing", *3GPP TS 36.141*,

[92] R. W. Heath Jr and A. Lozano, *Foundations of MIMO communication*. Cambridge University Press, 2018.

[93] P. Viswanath and D. N. C. Tse, "Sum capacity of the vector Gaussian broadcast channel and uplink-downlink duality", vol. 49, no. 8, pp. 1912–1921, 2003.

[94] 3GPP, "Study on channel model for frequencies from 0.5 to 100 GHz", 3rd Generation Partnership Project (3GPP), Technical Report (TR) 38.901, Mar. 2017, Version 14.2.2. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3173.

[95] "IEEE Standard for Information technology - Local and metropolitan area networks - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications ", *IEEE Std 802.11n-2009*,

[96] J. Mitola, "Cognitive Radio", Ph.D. dissertation, Institutionen för teleinformatik, 2000.