Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia

COPPE UFRJ

GREEDY ALGORITHMS AND MACHINE LEARNING FOR COMMUNICATIONS

Marcele Oliveira Kuhfuss de Mendonça

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia Elétrica.

Orientador: Paulo Sergio Ramirez Diniz

Rio de Janeiro
Outubro de 2022

GREEDY ALGORITHMS AND MACHINE LEARNING FOR COMMUNICATIONS

Marcele Oliveira Kuhfuss de Mendonça

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

RIO DE JANEIRO, RJ – BRASIL
OUTUBRO DE 2022

Mendonça, Marcele Oliveira Kuhfuss de

Greedy algorithms and Machine Learning for communications/Marcele Oliveira Kuhfuss de Mendonça. – Rio de Janeiro: UFRJ/COPPE, 2022.

XXI, 143 p.: il.; 29,7cm.

Orientador: Paulo Sergio Ramirez Diniz

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2022.

Referências Bibliográficas: p. 129 – 143.

1. massive MIMO. 2. antenna selection. 3. matching pursuit. 4. machine learning. 5. deep learning. 6. OFDM system. I. Diniz, Paulo Sergio Ramirez. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

To my family

Agradecimentos

Agradeço à minha família querida: meus pais Maria Cláudia e Marcelo e meu irmão Marcel. Sem vocês não teria conseguido chegar até aqui. Vocês são tudo para mim. Mesmo estando distante de vocês uma parte deste doutorado, sempre ficamos unidos e eu sentia o apoio e suporte de vocês. Mãe, obrigada, por ser essa mãe e melhor amiga maravilhosa. Obrigada por cuidar de mim e me colocar pra cima nos momentos difíceis (que não foram poucos). Pai, obrigada, por ser não só um pai, mas um grande amigo e estar sempre presente. Obrigada por trazer alegria e encorajamento quando me sentia perdida e desmotivada. Marcel, obrigada, por seu apoio e por ser um irmão companheiro e grande amigo que sei que sempre posso contar. Agradeço também a meu irmão Matheus que está longe, mas envia boas energias. Agradeço ao meu tio Luiz Carlos e meus avós Antônia e Raimundo. Obrigada, tio, pela sua animação e por sempre poder contar contigo. Obrigada, vovó, por sempre me incentivar e por também sempre estar disposta a ajudar.

Agradeço ao meu orientador e segundo pai, Diniz. Obrigada por sempre acreditar que eu era capaz. Obrigada pela sua paciência e gentileza. Obrigada por tudo que me ensinou até aqui e por me tornar uma profissional um pouco melhor desde o mestrado. Obrigada por ser um orientador parceiro que veste a camisa do time. Obrigada por me apoiar nas decisões da minha carreira. Obrigada por ser um grande amigo sempre disposto a me ouvir e a dar ótimos conselhos.

Agradeço a meu amigo e orientador de mestrado e graduação, Tadeu Ferreira. Obrigada, Tadeu, por me ajudar em tudo que pode. Você sabe que minha lista de agradecimentos a você é bem longa. Obrigada por sua parceria nos artigos. Obrigada por sempre estar disponível para uma boa conversa. Obrigada pelos seus conselhos. Obrigada por sua amizade, ela é muito importante para mim.

Agradeço ao meu irmão de orientação, Jonathas. Obrigada por toda a sua ajuda neste trabalho e por ser um ótimo amigo.

Agradeço ao professor Pascal por sua orientação enquanto estive no seu laboratório na EPFL (Swiss Federal Institute of Technology Lausanne).

Agradeço a meu amigo e parceiro de artigos Javier por me dar o suporte necessário para concluir os projetos na EPFL.

Agradeço aos meus colegas da EPFL e do SMT por tornarem os meus dias mais leves com ótimas conversas durante cafés e almoços.

Agradeço aos professores Marcello, Eduardo, Raimundo e Apolinário, por aceitarem fazer parte da minha banca de doutorado.

Agradeço também ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo suporte financeiro durante o doutorado.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

ALGORITMOS GREEDY E APRENDIZADO DE MÁQUINA PARA COMUNICAÇÕES

Marcele Oliveira Kuhfuss de Mendonça

Outubro/2022

Orientador: Paulo Sergio Ramirez Diniz

Programa: Engenharia Elétrica

A estrada para o 6G é repleta de desafios. Com a proliferação de serviços *wireless*, a demanda por espectro disponível também cresce. Com o potencial do aprendizado de máquina, prevê-se que o 6G revolucione as comunicações sem fio conectando inteligência em vez de apenas coisas. No entanto, a utilização de técnicas de aprendizado de máquina tem limitações, como exigir muitos dados de treinamento e ser suscetível a perturbações deliberadas conhecidas como amostras adversárias. Aqui, descrevemos um método de seleção de dados para reduzir o tempo de treinamento de redes neurais, levando em consideração as amostras de dados mais relevantes. Além disso, tecnologias-chave conhecidas como *massive* MIMO e OFDM são possíveis soluções para lidar com a demanda por mais recursos para cobrir um número crescente de usuários, pois essas tecnologias podem aliviar a esperada escassez espectral. No entanto, *massive* MIMO e OFDM possuem suas próprias desvantagens, como o custo da construção da estação base, desperdício de espectro e distorções não lineares. Para reduzir o custo da estação base, este trabalho propõe três estratégias de seleção de antenas de baixa complexidade para o downlink de um sistema *massive* MIMO, visando reduzir o potência de transmissão. A estratégia proposta para selecionar as antenas é inspirada na técnica de *matching pursuit.* Para lidar com o desperdício de espectro em sistemas OFDM sem fio, também propomos duas redes neurais para melhorar a recepção e reduzir a quantidade de redundância necessária. Como o canal de comunicação sem fio é aberto e exposto, propomos uma estrutura de treinamento com amostras adversárias para lidar com ataques do tipo *jamming.*

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

GREEDY ALGORITHMS AND MACHINE LEARNING FOR COMMUNICATIONS

Marcele Oliveira Kuhfuss de Mendonça

October/2022

Advisor: Paulo Sergio Ramirez Diniz

Department: Electrical Engineering

To pave the road to 6G, many challenges should be surpassed. As wireless services proliferate, the demand for available spectrum also grows. With the potential of machine learning, 6G is predicted to transform wireless communications by connecting intelligence rather than just things. However, utilizing machine learning techniques has limitations, such as requiring a lot of training data and being susceptible to deliberate perturbations known as adversarial samples. Here, we describe a data selection method for reducing the training time of neural networks by considering the most relevant data samples. Moreover, key technologies known as massive MIMO and OFDM are possible solutions to deal with the demand for more resources to cover a growing number of users. These technologies may alleviate the expected spectral shortage. Massive MIMO and OFDM, however, have disadvantages, such as the cost of base station construction, spectrum waste, and nonlinear distortions. To reduce the base station cost, this work proposes three low complexity antenna selection strategies for the downlink of a massive MIMO system, aiming at reducing the transmission power. The matching pursuit technique inspires the proposed strategy to select the antennas. To address spectrum waste in wireless OFDM systems, we propose two neural networks to enhance the reception while reducing the required redundancy. Finally, as the wireless communication channel is open and exposed, we propose an adversarial training framework to deal with jamming attacks.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over the past few years, wireless communications systems have experienced significant advancement. Based on analog technology with FM modulation, the first generation (1G) analog cellular networks introduced mobile voice-calling services in the 1980s. Around 1990, the second generation (2G) digital cellular networks replaced the 1G analog systems using time-division or frequency-division techniques. Among the proposed systems, the Global System for Mobile Communications (GSM) offered transparent communication due to its open standard. The third generation (3G) systems represented by WCDMA, CDMA2000, and TD-SCDMA were created in 2001 to support high-speed data ranging from 144 kbps to 2 Mbps. These systems exploit the code-division multiple access (CDMA). In 2009, the fourth generation (4G) or Long Term Evolution (LTE) started to provide more support to broadband data by combining multiple-input multiple-output (MIMO) and orthogonal frequency division multiple access (OFDMA).

Instead of simply enhancing network capacities as in previous generations, the fifth generation (5G) systems extend mobile communication services to connect not only people but also things. As a result, various applications like virtual reality (VR), Internet of Things (IoT), automobiles, smart cities, and wereable devices are currently available. Though 5G is still in the process of being deployed globally, as mobile traffic grows and new users swiftly join, it will inevitably run into technical issues. Therefore, researchers are already investigating solutions to meet the future demands for data rates in 2030, moving towards the next generation of wireless technology, the sixth-generation (6G).

It is expected that 6G will revolutionize wireless communications by connecting intelligence rather than connecting things [6] with the power of machine learning (ML). Traditional ML-based frameworks, however, suffer from serious privacy issues as training is performed in a central server. On the other hand, decentralized ML solutions which maintains all personal information for training devices locally are becoming more suitable for 6G. Federated Learning (FL), for instance, is a potential decentralized ML technique approach where cooperating devices jointly train a shared model using their own personal data, and only transmit model updates instead of raw data to the central server [7]. Although we do not consider a decentralized architecture in this work, it can be easily applied in the ML solutions proposed in Chapter 5. We discuss it further in Chapter 7 when pointing future research directions.

Widely used in 5G communication systems, massive MIMO and OFDM may be viable solutions for future-generation wireless systems like beyond 5G and 6G [8, 9].

With an extensive array of antenna elements, massive MIMO systems can substantially increase spectral efficiency using simple linear processing. As the number of antennas is enormous compared to the number of served users in the cell, we can benefit from channel hardening where all small-scale effects vanish due to the favorable action of the law of large numbers [10]. Simple linear processing is allowed due to the quasi-orthogonal channels between each base station (BS) and the set of active users. Nonetheless, implementing massive MIMO systems in practice is a challenging task.

One of the challenges of realizing massive MIMO is the BS fabrication cost. Since radio frequency (RF) elements are required for each antenna, growing the number of antennas increases these elements and the BS cost. Motivated by this issue, we aim to select the most effective BS antenna signals to achieve a certain performance level at the receivers, leading to power savings. The number of required RF chains can also be reduced from M to S by using a network of RF switches to connect the RF chains with the subset of selected antennas. Hence, we can alleviate the BS cost and benefit from significant diversity gain. On the other hand, solving the problem of selecting S out of M available antennas by verifying all possible choices is quite a challenge. This problem can be solved via convex optimization, as shown in [3, 5], but this solution leads to high computational costs. Therefore, we propose an efficient way to solve the antenna selection problem by using a greedy algorithm called matching pursuit [11, 12], in which the computational cost is substantially reduced.

OFDM is a suitable modulation waveform for sixth-generation (6G) networks as it can be merged with other technologies [13]. OFDM offers many benefits, including the ability to combat the interference imposed by a multipath fading environment. By adding L redundant elements to the OFDM symbol, OFDM can mitigate the inter-symbol interference (ISI) and inter-block interference (IBI) caused by the multiple delayed versions of the transmitted signal [14]. The value of L depends on the maximum delay spread of the channel [15]. However, OFDM has significant challenges, such as spectrum waste due to overhead of the redundancy length and nonlinear distortion caused by techniques to reduce the peak-to-average power ratio (PAPR). In addition, when the channel order is not known in some practical cases, choosing the suitable amount of redundancy is also tricky. New solutions based on ML and deep learning (DL) techniques are then envisioned to address these challenges, unveiling promising possibilities for OFDM in the next communication generation [16].

Nevertheless, using machine learning techniques has drawbacks, such as needing a large amount of training data and being vulnerable to malicious perturbations known as adversarial examples. Adversarial examples are quasi-imperceptible perturbed versions of the original samples designed to trick neural networks. These perturbations of the network input can lead to disastrous implications in critical areas where wrong decisions can directly affect human lives. Adversarial training is the most efficient solution to defend the network against these malicious attacks. However, adversarial trained networks generally come with lower clean accuracy and higher computational complexity. Therefore, designing robust machine learning solutions while reducing computational costs is essential. In this work, we present a data selection (DS) strategy to consider the most relevant data samples during the training of a standard neural network while reducing its training time. The DS technique is also applied during the training of an adversarial neural network in an attempt to increase the accuracy to unseen adversarial examples while not reducing the accuracy to unseen clean samples.

Regarding the main drawbacks in OFDM systems, we propose two neural networks to enhance the OFDM reception in the presence of transmitter nonlinearity and insufficient redundancy. The first neural network is responsible for improving the least square (LS) channel estimation, whereas the second one refines symbol detection. We also propose a reinforcement learning approach to choose the amount of redundancy based on the available mean squared error (MSE) between the estimated channel and the true channel.

Due to the open and exposed nature of the wireless communication channel, wireless OFDM systems are also susceptible to jamming attacks. These harmful attacks have the potential to impair communication seriously. As transmitting pilot symbols plays a significant role in channel estimation and equalization, pilot jamming is often used to degrade the system performance. Although neural networks (NNs) can be used to improve channel estimation, they are also vulnerable to adversarial examples. Therefore, applying machine learning-based approaches in communications systems raises the risk. Adversarial examples are extensively explored in classification problems where the NN output prediction is flipped. However, provided the requirements are sufficiently described, adversarial examples can also be considered in regression tasks like channel estimation. We then introduce an adversarial training framework adapted to OFDM systems to provide robust and reliable NN solutions.

1.1 Organization

The structure of this work is as follows. In Chapter 2, we study the antenna selection problem in the downlink of a massive MIMO system. We propose three low complexity selection schemes performed whenever a symbol is ready for transmission [17]. In Chapter 3, we investigate machine learning methods to improve wireless OFDM reception, focusing on supervised and semi-supervised learning approaches. We also discuss adversarial training to develop robust ML-based solutions. Chapter 4 presents a data selection strategy we can apply during standard [18, 19] and adversarial training [20]. The machine learning-based solutions for practical problems in wireless OFDM systems are proposed in Chapter 5. The solutions comprise choosing the proper amount of redundancy and improving the performance when insufficient redundancy is employed [21–23]. We also propose designing systems robust to jamming attacks in Chapter 6. Some concluding remarks are included in Chapter 7, along with possible future work.

1.2 List of Publications

This work is based on the following research publications.

- [17]: MENDONÇA, M. O. K., DINIZ, P. S. R., FERREIRA, T. N., LOVISOLO, L. Antenna selection in massive MIMO based on greedy algorithms. IEEE Transactions on Wireless Communications, v. 19, n. 3, p. 1868-1881, 2019 (Chapter 2).

- [24]: MENDONÇA, M. O. K., NETTO, S. L., DINIZ, P. S. R., THEODORIDIS, S., Machine Learning, Chapter 13, on Signal Processing and Machine Learning Theory edited by DINIZ, P. S. R. Academic Press, Cambridge, UK, 2022. (Chapter 3).

- [18]: MENDONÇA, M. O. K; FERREIRA, J. O.; DINIZ, P. S. R. Data Selective Deep Neural Networks for Image Classification. In: 2021 29th European Signal Processing Conference (EUSIPCO). IEEE, 2021. p. 1376-1380 (Chapter 4).

- [19]: FERREIRA, J. O.; MENDONÇA, M. O. K.; DINIZ, P. S. R. Data selection in neural networks. IEEE Open Journal of Signal Processing, v. 2, p. 522-534, 2021 (Chapter 4).

- [20]: MENDONÇA, M. O. K., DINIZ, P. S. R., MAROTO, J., FROSSARD, P. Adversarial training with informed data selection, accepted to the European Signal Processing Conference (EUSIPCO), 2022 (Chapter 4).

- [21]: MENDONÇA, M. O. K., DINIZ, P. OFDM receiver using deep learning: Redundancy issues. In: 2020 28th European Signal Processing Conference (EUSIPCO), pp. 1687–1691. IEEE, 2021 (Chapter 5).

- [22]: DINIZ, P. S. R., MENDONÇA, M. O. Zero-Padding OFDM Receiver Using Machine Learning. In: 2021 IEEE Statistical Signal Processing Workshop (SSP), pp. 26–30. IEEE, 2021 (Chapter 5).

- [23]: MENDONÇA, M. O. K., DINIZ, P. S. R., FERREIRA, T. N. Machine learning-based channel estimation for insufficient redundancy OFDM receivers using comb-type pilot arrangement, accepted to Latin-American Conference on Communications (LATINCOM), 2022 (Chapter 5).

1.3 Notation

Vectors and matrices are represented by characters in bold type in which lower-case letters are used for vectors and upper-case letters for matrices, whereas non-bold letters are scalar variables. We consider column vectors, and a_m represents the mth component of vector $\mathbf{a}$. In a similar way, the entries of a matrix $\mathbf{A}$ are of the form a_{mk} in which m represents the row and k the column of $\mathbf{A}$. In order to identify a column of a matrix, we represent it as a column vector $\mathbf{a}_k$ where k is the column index. For example, an $M \times K$ matrix $\mathbf{A}$ can be written as

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1K} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1} & a_{M2} & \dots & a_{MK} \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_K \end{bmatrix}. \tag{1.1}$$

When the elements of a vector are random variables, we represent the vector by a character in bold italic type, i.e., $\boldsymbol{a}$. The statement $\boldsymbol{a} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ means that the random vector $\boldsymbol{a}$ is distributed as a real Gaussian random variable with zero mean and covariance matrix $\mathbf{R} = \mathrm{E}(\mathbf{a}\mathbf{a}^{\mathrm{T}})$. The distribution in the statement $\boldsymbol{a} \sim \mathcal{CN}(\mathbf{0}, \mathbf{R})$ is known as circularly simmetric complex Gaussian, which means that $e^{j\phi}\boldsymbol{a} \sim \mathcal{CN}(\mathbf{0}, \mathbf{R})$ for any given ϕ. $\mathrm{E}[\cdot]$ and $\mathrm{Var}[\cdot]$ denote the expected value and variance of a random variable, respectively.

In general, we use subscripts in vectors and matrices just to represent the variable's name. However, subscripts in parentheses refer to the size of a square matrix. For example, $\mathbf{I}_{(K)}$ is the identity matrix with size $K \times K$.

The real, complex, natural, and integer sets are represented by the following symbols $\mathbb{R}$, $\mathbb{C}$, $\mathbb{N}$ and $\mathbb{Z}$. For example, we can establish that matrix $\mathbf{A} \in \mathbb{R}^{M \times K}$ and $\mathbf{a}_K \in \mathbb{R}^{M \times 1}$ in equation (1.1).

The operators used throughout the text are organized in Table 1.1.

Table 1.1: Operators used throughout this work

Operator	**Input**	**Output**
$(\cdot)^{\mathrm{T}}$	vector or matrix	input vector or matrix with transposed elements
$(\cdot)^{\mathrm{H}}$	vector or matrix	input vector or matrix with transposed and conjugated elements
$(\cdot)^{*}$	vector or matrix	vector or matrix of the complex conjugate elements of the input vector or matrix
$(\cdot)^{-1}$	matrix	inverse of input matrix
$\lVert\mathbf{x}\rVert_p$	vector	p-norm, $\left(\sum_{m=1}^{M} \lvert x_m\rvert^p\right)^{1/p}$
$\lVert\cdot\rVert_0$	vector	number of non-zero entries of input vector
$\mathrm{diag}\,(\cdot)$	vector	diagonal matrix where the diagonal entries are the elements of the input vector
$\det(\cdot)$	square matrix	determinant of the input matrix
$\mathrm{tr}(\cdot)$	square matrix	trace of the matrix, that is, the sum of the diagonal elements of the input matrix
$\langle\cdot,\cdot\rangle$	two vectors	inner product between two input vectors
$\mathrm{vec}(\cdot)$	matrix	column vector which is obtained by transposing the rows of the input matrix and stacking them up
$\mathrm{rem}(\cdot)$	matrix	matrix with zero columns removed

Chapter 2

Antenna selection in massive MIMO based on greedy algorithms

2.1 Introduction

Massive MIMO is one of the most promising technologies for the future-generation wireless systems for improving power and spectral efficiency and, hence, meet the users' demands for higher data rates [25–30]. In such a scenario, the BS is equipped with a large number of antennas M to serve a set of K terminals or users' equipment sharing the same time-frequency resource. Since multiple data streams are simultaneously transmitted, precoding techniques are well motivated to reduce multiuser interference [31]. In massive MIMO, one considers that $M \gg K$, which brings about the favorable action of the law of large numbers [10]. This effect is known as channel hardening, in which the small-scale fading effects vanish as the number of antennas increases. Moreover, simple linear processing is possible due to the quasi-orthogonal nature of the channels between each BS and the set of active users [29, 32].

Each MIMO system antenna typically has its dedicated chain of radio frequency (RF) elements. As a result, adding more antennas results in more RF chains. Since each RF chain contains costly components, such as high-resolution digital-to-analog converters (DAC) and amplifiers, allocating a dedicated RF chain to each antenna is practically infeasible. One way to reduce the number of RF chains is to employ a network of RF switches that connect the RF chains with the most effective subset of S-selected BS antennas [33, 34].

Basically, antenna selection can be performed before the precoding stage or both procedures can work as a unit. To distinguish these approaches, we refer to the first one as an antenna selection and the latter as a precoding-antenna selection.

Precoding consists of modifying the stream of symbols before transmission in order to minimize the effects of channel distortion, noise, and multiuser interference [35]. Prior knowledge regarding the channel-state-information (CSI) and/or the symbols is used to compute the new vector to be transmitted, and it is also common to classify the precoder as either channel/block-level or symbol–level [36]. A channel–level precoder is only dependent on the channel estimates, whereas a symbol–level precoder is dependent on both the channel estimates and the symbols. Similarly, the antenna selection method can also be classified as channel–level or symbol–level. A channel–level antenna selector chooses a new subset of activated antennas when CSI is available at BS, whereas a symbol–level antenna selector chooses such a subset whenever a stream of symbols is available for transmission.

Solving the problem of selecting S out of M available antennas by verifying all possible choices becomes prohibitive as M increases. In the channel–level context, one of the criteria used to select the antennas is maximizing the downlink capacity [3, 37–39] considering fixed user power allocation. In [3], the binary constraints, which determine if an antenna is active or not, are replaced by convex constraints that enable a convex relaxed optimization problem. Such an optimization problem can be solved by using interior-point methods [40, 41], and the subset of selected antennas is obtained using a rounding step. In [42], the authors obtain a subset of antennas by finding a channel submatrix with the largest minimum singular value. The sum rate is another common criterion employed to select the active antennas [43–46]. Moreover, the user scheduling problem can be jointly addressed as proposed in [44, 46]. In the symbol–level context, recent methods exploit the sparse recovery problem in the precoding stage so that antenna selection and precoding procedures are jointly performed [5, 33].

The antenna selection problem can be solved via convex optimization, as shown in [3] and [5], but the solution involves high computational costs at the BS. As a consequence, low complexity antenna selection approaches have been developed in order to minimize resource usage. For example, random antenna selection and power/SNR (signal-to-noise ratio) based antenna selection are simple channel–level methods that achieve reasonable performance.

In this chapter, we develop antenna selection methods based on a greedy algorithm called matching pursuit (MP) [11], in which the computational costs are substantially reduced. First, we describe the considered massive MIMO system and the antenna selection problem in Section 2.2. Then, in Section 2.3, we formulate the antenna selection using the MP framework and introduce the matching pursuit generalized bit planes (MPGBP) algorithm [47] and propose a complex version. Moreover, we propose a novel residue rate of decay to cope with the antenna selection and summarize the three groups of antenna selectors considered in this work. We advance a channel–level algorithm suitable for selecting the antennas in slowly-varying fading channels in Section 2.4. When CSI is available, the channel responses for the BS to all the terminals in the cell are utilized to perform the antenna selection. In this case, we describe an alternative description of the approximation problem to allow the use of the MP formulation. Although the performance of the presented channel–level algorithm is not justified by its complexity, its derivation was a starting point for the proposed algorithms. Then, in order to handle scenarios with short coherence times, we propose three symbol-level techniques in Sections 2.5 and 2.6. In this design, the channel responses and the transmitted message are considered to create the subset of active antennas. The symbol–level algorithms require restraining the residue energy decay rate so that enough antennas are selected – i.e., one must balance sparsity (MP) and diversity (MIMO). Two of the symbol–level methods consider precoding while selecting the antennas so that both operations are jointly performed, reducing the computational complexity since no matrix inversion is required. Moreover, one of the latter algorithms is based on the matching pursuit generalized bit planes (MPGBP), which obtains a quantized-element message, alleviating the amplifier linearity demands and further reducing the BS cost. In Section 2.7, we provide the complexity of each proposed algorithm. Section 2.8 evaluates the MP-based methods in several scenarios. Finally, we end the chapter with some concluding remarks in Section 2.9.

2.2 System model

We consider a downlink massive MIMO system in a single cell as illustrated in Figure 2.1. The base station is equipped with M antennas and simultaneously serves K single antenna terminals.

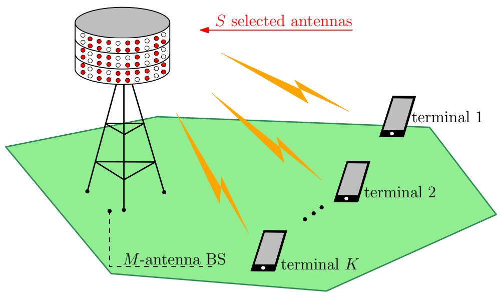


Figure 2.1: Downlink massive MIMO with only S selected BS-antennas.

When all the M antennas are active, the BS transmits the signal $\mathbf{x} \in \mathbb{C}^{M\times 1}$ to the terminals in the cell. In order to mitigate the user interference, a precoding technique can be employed so that vector $\mathbf{x}$ is a precoded version of the vector $\mathbf{q} \in \mathbb{C}^{K\times 1}$ containing the symbols intended for each terminal. If a linear precoding scheme is applied, the transmitted vector can be written as

$$\mathbf{x} = \mathbf{P}\mathrm{diag}\left(P_T\boldsymbol{\eta}\right)^{1/2}\mathbf{q}, \tag{2.1}$$

where $\mathbf{P} \in \mathbb{C}^{M\times K}$ is the precoding matrix, and $\mathrm{diag}\left(P_T\boldsymbol{\eta}\right)^{1/2}$ is the diagonal matrix with elements $\sqrt{P_T\eta_1}$,$\sqrt{P_T\eta_2}, \ldots, \sqrt{P_T\eta_K}$. The power allocated for each terminal is represented by $P_T\eta_k$ and

$$\sum_{k=1}^{K} P_T\eta_k = P_T, \tag{2.2}$$

where P_T is the transmission power and η_k is the power control coefficient for terminal k. Therefore the received signal vector $\mathbf{y} \in \mathbb{C}^{K\times 1}$ of all the terminals is

$$\mathbf{y} = \mathbf{G}^{\mathrm{T}}\mathbf{x} + \mathbf{w}, \tag{2.3}$$

in which $\mathbf{G} \in \mathbb{C}^{M\times K}$ is the matrix comprising the channel responses for the BS to all the terminals in the cell. Each element w_k of vector $\mathbf{w} \in \mathbb{C}^{K\times 1}$ is a realization of a circularly symmetric complex Gaussian distribution $\mathcal{CN}(0, \sigma_w^2)$ random variable representing noise.

Terminal k receives the scaled symbol of interest q_k plus additive interference and noise

$$\begin{aligned} y_k &= \mathbf{g}_k^{\mathrm{T}}\mathbf{x} + w_k \\ &= \underbrace{\mathbf{g}_k^{\mathrm{T}}\mathbf{p}_k\sqrt{\eta_k}q_k}_{\text{Desired signal}} + \underbrace{\mathbf{g}_k^{\mathrm{T}}\sum_{\substack{k'=1\\k'\neq k}}^{K}\mathbf{p}_{k'}\sqrt{\eta_{k'}}q_{k'}}_{\text{User-interference}} + \underbrace{w_k}_{\text{Noise}}, \end{aligned} \tag{2.4}$$

where $\mathbf{g}_k$ and $\mathbf{p}_k \in \mathbb{C}^{M\times 1}$ are the k-th column of $\mathbf{G}$ and $\mathbf{P}$, respectively [48].

Zero-forcing (ZF) [49] and maximum ratio (MR) [50] are examples of linear precoding in which the precoding matrix is $\mathbf{P} = \mathbf{G}^*(\mathbf{G}^{\mathrm{T}}\mathbf{G}^*)^{-1}$ for ZF and $\mathbf{P} = \mathbf{G}^*$ for MR. Assuming channel reciprocity, the BS obtains CSI in the uplink and hence computes the correspondent precoding matrix. We consider two possible propagation environments. In the first one, the system operates in a rich scattering environment with non-line-of-sight (NLOS) propagation so that the signal arrives at BS through many independent identically distributed (i.i.d.) paths. Hence, the uncorrelated Rayleigh fading [48] model can be used to model $\mathbf{g}_k$ as a realization of the random variable

$$\boldsymbol{g}_k \sim \mathcal{CN}(\mathbf{0}_{(M)},\, \beta_k\mathbf{I}_{(M)}). \tag{2.5}$$

Consequently, g_{mk} can be modeled as

$$g_{mk} = \sqrt{\beta_k} h_{mk} \tag{2.6}$$

where β_k is a large-scale coefficient dependent only on k and h_{mk} is the realization of a random variable distributed as $\mathcal{CN}(0,\, 1)$, which represents the effect of small-scale fading. In the second case, the BS receives the signal only through a small number of significant paths N, whereas space-selective fading is observed [51]. For example, we consider the case that the BS is a uniform linear array (ULA) with antenna spacing d_H, and the terminals are located at fixed locations in the far field of the array. Then, each of the multipath components results in a plane wave that reaches the antenna from a particular angle θ_{kn} and gives an array response or steering vector $\mathbf{s}_{kn} \in \mathbb{C}^{M \times 1}$

$$\mathbf{s}_{kn} = h_{kn} \begin{bmatrix} 1 & e^{2\pi \mathrm{j} d_H \sin(\theta_{kn})} \dots e^{2\pi \mathrm{j} d_H (M-1) \sin(\theta_{kn})} \end{bmatrix}^{\mathrm{T}} \tag{2.7}$$

where $h_{kn} \in \mathbb{C}$ accounts for the gain and phase-rotation for this path. Considering channel reciprocity between the transmitter and receiver, the channel response for terminal k is the superposition

$$\mathbf{g}_k = \sum_{n=1}^{N} \mathbf{s}_{kn} \tag{2.8}$$

of the array responses of the N-path components. In this way, the BS obtains the CSI of the k-th terminal to the BS.

2.2.1 Antenna selection

To reduce the number of RF chains, we select S out of M BS antennas, leading to exploitation of redundancy in the spatial diversity while saving power as fewer antennas are activated. In this way, the BS transmits the reduced message

$$\mathbf{x}_S \in \mathbb{C}^{M \times 1}, \|\mathbf{x}\|_0 = S, \tag{2.9}$$

where the subscript S indicates that $\mathbf{x}$ has S non-zero elements. The simpler approach is to obtain the reduced message $\mathbf{x}_S$ by employing a linear precoding scheme in which the precoding matrix is computed considering a selection rule. In fact, the antenna selection algorithm produces the selection vector

$$\mathbf{z} = [z_1 \; z_2 \dots z_M]^{\mathrm{T}} \in \{0{,}1\}^M \tag{2.10}$$

which must satisfy $\mathbf{1}^{\mathrm{T}}\mathbf{z} = S$. After the selection is performed, the S-selected channel matrix

$$\mathbf{G}_S = \mathrm{diag}\left(\mathbf{z}\right) \mathbf{G}, \tag{2.11}$$

is computed, and an external precoding scheme should be employed to produce $\mathbf{x}$. If a linear precoding is chosen, the vector to be transmitted can be computed as

$$\mathbf{x}_S = \mathbf{P}_S \mathrm{diag}\left(P_T \boldsymbol{\eta}\right)^{1/2} \mathbf{q}, \tag{2.12}$$

where $\mathbf{P}_S$ is the precoding matrix. For example, the choice can be $\mathbf{P}_S = \mathbf{G}_S^*(\mathbf{G}_S^{\mathrm{T}}\mathbf{G}_S^*)^{-1}$ with $S \geq K$ for ZF precoding or $\mathbf{P}_S = \mathbf{G}_S^*$ for MR precoding. In this case, the antenna selection algorithm only points out the antennas that should be active, and a precoding technique should be subsequently employed.

Alternatively, the antenna selection algorithm could jointly select the antenna and the correspondent precoding by solving the problem of finding the vector $\mathbf{x}$ that best represents the symbol vector $\mathbf{q}$ depending on the channel matrix. Hence, $\mathbf{x}$ is directly obtained. Finally, we present algorithms to perform antenna selection and precoding using dictionary based approximations, which will be further explained in the next section.

2.3 Matching pursuit antenna selection

Matching pursuit (MP) is a greedy algorithm employed to represent a signal using a redundant dictionary [11, 12, 52]. Iteratively, the MP aims to solve

$$\begin{aligned} \underset{\mathbf{a}\in\mathbb{C}^{M\times 1}}{\text{minimize}} \quad & \|\mathbf{Da}-\mathbf{b}\|_2^2 \\ \text{subject to} \quad & \|\mathbf{a}\|_0 \leq S \end{aligned} \tag{2.13}$$

by approximating the target vector $\mathbf{b} \in \mathbb{C}^{K\times 1}$ using the dictionary matrix $\mathbf{D} \in \mathbb{C}^{K\times M}$ and a sparse vector $\mathbf{a} \in \mathbb{C}^{M\times 1}$. The columns of the dictionary $\mathbf{D} = [\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_M]$ are called codewords, and they are used to represent the target vector $\mathbf{b}$.

Originally, the MP method aimed at finding the sparsest vector $\mathbf{a}$, and there is no constraint regarding selecting a codeword more than once. However, in the antenna selection problem, we are interested in selecting a fixed number of antennas, S. Then it is more convenient to impose that a codeword is chosen only once, as the solution of the following optimization problem

$$\begin{aligned} \underset{\mathbf{a}\in\mathbb{C}^{M\times 1}}{\text{minimize}} \quad & \|\mathbf{Da}-\mathbf{b}\|_2^2 \\ \text{subject to} \quad & \|\mathbf{a}\|_0 = S. \end{aligned} \tag{2.14}$$

Instead of approximating the l_0-norm constrained problem and applying a sophisticated optimization method or starting an exhaustive search over all possible combinations, the MP algorithm tries to find the best solution at each iteration. As a matter of fact, it searches for the codeword $\mathbf{c}_m, m = 1, \cdots M$ which is closest to the current approximation residue. The inner product between two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{C}^N$

$$\langle \mathbf{x},\mathbf{y} \rangle = \sum_{i=1}^{N} x_i^* y_i = \mathbf{x}^{\mathrm{H}}\mathbf{y} \tag{2.15}$$

is used to measure how close the residue is to the codewords. In this way, the codeword with the largest inner product is selected at iteration i, and it is represented by $\mathbf{c}_{m_i}$.

In the first iteration, the residue is the target vector, whereas in the remaining iterations it is composed by

$$\mathbf{r}_i = \mathbf{r}_{i-1} - \gamma \langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i} \rangle \mathbf{c}_{m_i}, \tag{2.16}$$

that is, the last residue $\mathbf{r}_{i-1}$ subtracted from the selected codeword $\mathbf{c}_{m_i}$ scaled by its correlation with the residue $\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i} \rangle$ and a scale factor γ. In this way, as the number of iterations of the MP algorithm increases, the residue norm decreases. The approximate version of $\mathbf{b}$

$$\widehat{\mathbf{b}}_I = \sum_{i=1}^{I} \langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle \mathbf{c}_{m_i} \tag{2.17}$$

is composed of the sum of the projections of the current residue $\mathbf{r}_i$ on the selected codeword $\mathbf{c}_{m_i}$, and $I = S$ is the total number of iterations, selecting one antenna per iteration.

2.3.1 Matching pursuit with generalized bit planes

It is desirable in practical applications that $\langle \mathbf{r}_i, \mathbf{c}_{i_m} \rangle$ in equation (2.17) be quantized. The version of MP called matching pursuit with generalized bit planes (MPGBP) [12] imposes the approximation

$$\widehat{\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle} = \alpha^{\mu_i}, \mu_i \in \mathbb{Z}, \alpha < 1, \quad i \in [1,S] \tag{2.18}$$

where

$$\mu_i = \left\lceil \log_\alpha \left(\frac{2\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle}{1+\alpha} \right) \right\rceil, \quad i \in [1,S], \tag{2.19}$$

in which $\lceil z \rceil$ is the smallest integer larger than or equal to z. At iteration i, the current residue is approximated to a quantized level represented by α^{μ_i}. Then, the approximated vector is formed by a weighted linear combination of the matched codewords where the weights are the quantized levels.

2.3.2 Complex matching pursuit generalized bit planes

In the MPGBP algorithm [12], the approximation $\widehat{\mathbf{b}}$ of $\mathbf{b}$ is written in quantized form as

$$\widehat{\mathbf{b}} = \sum_{i=1}^{I} \alpha^{\mu_i} \mathbf{c}_{m_i}, \tag{2.20}$$

where $\alpha < 1$. To obtain the approximation $\alpha^{\mu_i} \approx \langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle$, μ_i is defined as

$$\mu_i = \left\lceil \log_\alpha \left(\frac{2\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle}{1+\alpha} \right) \right\rceil. \tag{2.21}$$

However, in the case that $\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle \in \mathbb{C}$, using different bit planes for the real and imaginary parts of the inner product is preferred. Therefore, the calculation of μ_i introduced in [12] is split in

$$\mu_{i_1} = \left\lceil \log_2 \left(\frac{3}{\mathrm{Re}\{\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle\}} \right) \right\rceil \tag{2.22}$$

and

$$\mu_{i_2} = \left\lceil \log_2 \left(\frac{3}{\mathrm{Im}\{\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle\}} \right) \right\rceil, \tag{2.23}$$

in which $\mathrm{Re}\{z\}$ and $\mathrm{Im}\{z\}$ take the real and imaginary part of z, respectively and $\alpha = 1/2$.

The resulting quantized form of $\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle$ is

$$\alpha^{\mu_i} = \frac{\mathrm{Re}\{\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle\}}{||\mathrm{Re}\{\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle\}||} \alpha^{\mathrm{Re}\{\mu_{i_1}\}} + \frac{\mathrm{Im}\{\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle\}}{||\mathrm{Im}\{\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle\}||} j\alpha^{\mathrm{Re}\{\mu_{i_2}\}}. \tag{2.24}$$

2.3.3 Dictionary normalization and residue rate of decay

In the MP, the columns of the dictionary matrix $\mathbf{D}$ are normalized. The residue is updated following the rule in equation (2.16),

$$\mathbf{r}_i = \mathbf{r}_{i-1} - \gamma \langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i} \rangle \mathbf{c}_{m_i}, \tag{2.25}$$

in which $\mathbf{c}_{m_i}$ is the codeword with maximum inner product and γ is originally equal to one. Since $\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i} \rangle = \mathbf{r}_{i-1}^{\mathrm{H}} \mathbf{c}_{m_i}$, by computing $\mathbf{r}_i^{\mathrm{H}} \mathbf{r}_i$, one obtains the residue energy

$$\begin{aligned} ||\mathbf{r}_i||^2 = \mathbf{r}_i^{\mathrm{H}} \mathbf{r}_i &= \mathbf{r}_{i-1}^{\mathrm{H}} \mathbf{r}_{i-1} - \gamma (\mathbf{r}_{i-1}^{\mathrm{H}} \mathbf{c}_{m_i})^2 - \gamma (\mathbf{r}_{i-1}^{\mathrm{H}} \mathbf{c}_{m_i})^* (\mathbf{c}_{m_i}^{\mathrm{H}} \mathbf{r}_{i-1}) \\ &+ \gamma^2 (\mathbf{r}_{i-1}^{\mathrm{H}} \mathbf{c}_{m_i})^* (\mathbf{r}_{i-1}^{\mathrm{H}} \mathbf{c}_{m_i}) \mathbf{c}_{m_i}^{\mathrm{H}} \mathbf{c}_{m_i} \\ &= ||\mathbf{r}_{i-1}||^2 - \gamma \{ (\mathbf{r}_{i-1}^{\mathrm{H}} \mathbf{c}_{m_i})^2 + [(\mathbf{r}_{i-1}^{\mathrm{H}} \mathbf{c}_{m_i})^*]^2 \} + \gamma^2 (\mathbf{r}_{i-1}^{\mathrm{H}} \mathbf{c}_{m_i})^* (\mathbf{r}_{i-1}^{\mathrm{H}} \mathbf{c}_{m_i}) ||\mathbf{c}_{m_i}||^2. \end{aligned} \tag{2.26}$$

Considering normalized codewords, $||\mathbf{c}_{m_i}||^2 = 1$, we have

$$\begin{aligned} ||\mathbf{r}_i||^2 &= ||\mathbf{r}_{i-1}||^2 - \gamma\{[2\mathrm{Re}(\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle)]^2 - [2\mathrm{Im}(\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle)]^2\} + \gamma^2|\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle|^2||\mathbf{c}_{m_i}||^2 \\ &= \left(1 - 2\gamma\frac{\{[\mathrm{Re}(\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle)]^2 - [\mathrm{Im}(\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle)]^2\}}{||\mathbf{r}_{i-1}||^2} + \gamma^2\frac{|\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle|^2}{||\mathbf{r}_{i-1}||^2}\right)||\mathbf{r}_{i-1}||^2. \end{aligned} \tag{2.27}$$

As $|\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle|^2 = [\mathrm{Re}(\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle)]^2 + [\mathrm{Im}(\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle)]^2$ then

$$[\mathrm{Re}(\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle)]^2 - [\mathrm{Im}(\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle)]^2 < |\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle|^2, \tag{2.28}$$

and hence equation (2.27) becomes

$$\begin{aligned} ||\mathbf{r}_i||^2 &< \left(1 - (2\gamma - \gamma^2)\frac{|\langle \mathbf{r}_{i-1}, \mathbf{c}_{m_i}\rangle|^2}{||\mathbf{r}_{i-1}||^2}\right)||\mathbf{r}_{i-1}||^2 \\ &< \left(1 - (2\gamma - \gamma^2)\delta(\mathbf{D},\mathbf{r}_{i-1})\right)||\mathbf{r}_{i-1}||^2, \end{aligned} \tag{2.29}$$

where $\delta(\mathbf{D},\mathbf{r}_{i-1})$ is the squared correlation ratio [52] or decay factor [53] at iteration i. Let $\delta(\mathbf{D}) = \inf_{\mathbf{r}_{i-1},\ i} \delta(\mathbf{D},\mathbf{r}_{i-1})$ [52–54], equation (2.29) becomes

$$||\mathbf{r}_i||^2 < \left(1 - (2\gamma - \gamma^2)\delta(\mathbf{D})\right)||\mathbf{r}_{i-1}||^2, \tag{2.30}$$

so that we can obtain a bound for the residue energy in a non-recursive form as

$$||\mathbf{r}_i||^2 < \left(1 - (2\gamma - \gamma^2)\delta(\mathbf{D})\right)^i||\mathbf{b}||^2, \tag{2.31}$$

where $\mathbf{r}_1 = \mathbf{b}$, that is, the initial residue is the target vector itself.

One can observe in equation (2.31) that if $\gamma = 1$ and the correlation ratio of signal $\mathbf{b}$ and the dictionary are high, the residue norm decreases quickly, and hence a sparse vector $\mathbf{a}$ is found. In the antenna selection problem, at each iteration i, an antenna with index m_i is selected. Thus, if we want to select S among M antennas, and the residue norm decays quickly, as illustrated in Figure 2.2 for $M = 400$, $S = 100$, $K = 12$ and $\gamma = 1$, one will actually select fewer antennas than desired. On the other hand, if the correlation ratio is low, the residue norm decreases slowly and it means that the information of signal $\mathbf{b}$ spreads across the selected codewords [52]. Therefore, one selects a sub-optimal factor γ, $0 < \gamma \leq 1$ [52], so that at iteration $i = S$, the energy of signal $\mathbf{b}$ is diluted across S codewords. In this case, the residue energy is bounded by

$$||\mathbf{r}_S||^2 < \left(1 - (2\gamma - \gamma^2)\delta(\mathbf{D})\right)^S||\mathbf{b}||^2. \tag{2.32}$$

This is illustrated in the curve for $\gamma \approx 0.04$ in Figure 2.2.

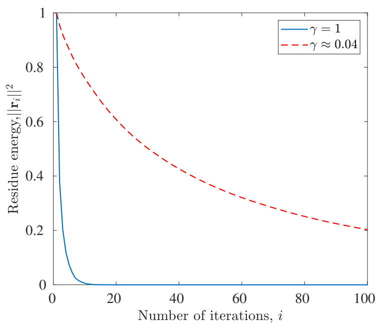


Figure 2.2: Decay of the energy of the residue for an MP problem with $\mathbf{D} \in \mathbb{C}^{K \times M}$, $\mathbf{a} \in \mathbb{C}^{M \times 1}$, $\mathbf{b} \in \mathbb{C}^{K \times 1}$, $M = 400$, $S = 100$, and $K = 12$.

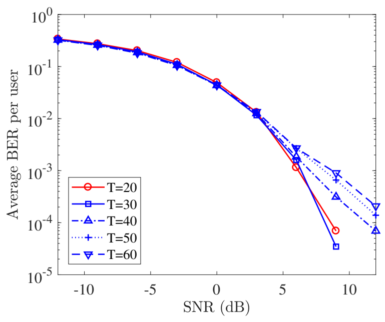
In the following proposed antenna selection methods, the dictionary matrix has unit-norm columns. The MP residue is updated using the rule in equation (2.25). This update rule slows down the residue norm decrease rates; nevertheless, more significant terms are included in the decomposition. In this way, the MP greediness is sacrificed to allow the selection of S antennas.

The sub-optimal factor is defined as

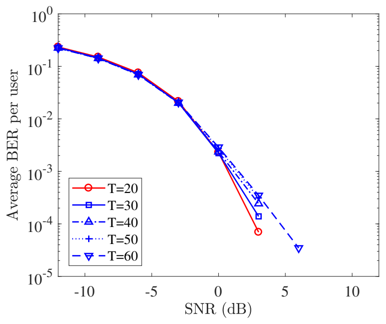
$$\gamma = \frac{K}{S}\left(\frac{K}{K+T}\right), \tag{2.33}$$

if $S > K$, whereas in the case that $S \leq K$, no adjustment needs to be made. It is worth mentioning that when performing precoding where $S \leq K$, the ZF solution does not exist. As it will be seen in the simulation results, the proposed methods with precoding obtain lower BER levels when compared with the methods that implement ZF precoding. In the experiments presented in Section 2.8, $T = 20$ is a suitable choice for the scenarios comprising perfect and partial CSI, rich and poor scattering, 4-QAM and 16 QAM, and ZF and MR precoding. This sub-optimal factor leads to good experimental results, and it is a reasonable choice since γ is proportional to the ratio of the codeword length and the desired quantity of antennas which are both parameters of the antenna selection problem.

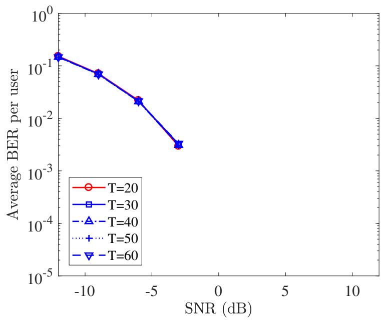
The sub-optimality factor γ is inversely proportional to the desired quantity of antennas. Indeed, more antennas to select means that the residue rate of decay should be lower. Although T is chosen experimentally, its value does not severely impact the BER, as shown in Figure 2.3 for the Symbol–Level Matching Pursuit Precoding Antenna Selection (SL-MPPAS) algorithm, which will be properly introduced in Subsection 2.6.1.



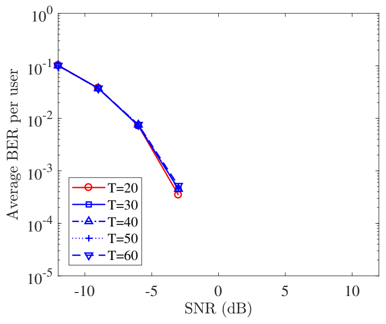
(a) $S = 12$.



(b) $S = 40$.



(c) $S = 100$.



(d) $S = 200$.

Figure 2.3: Average BER per user for the proposed Symbol–Level Matching Pursuit Precoding Antenna Selection (SL-MPPAS) algorithm when parameter T is varied for $S \in \{12, 440, 100, 200\}$ selected antennas. In this experiment, a base station with $M = 400$ antennas transmits 4-QAM symbols (precoded using ZF) to $K = 12$ terminals using only S antennas active. The massive MIMO system has perfect CSI knowledge and the channel is modelled considering a rich scattering environment.

2.3.4 Antenna selection problems

In this subsection, we summarize the antenna selection variants which will be further discussed in the following sections. We consider channel–level, symbol–level, and symbol–level precoding antenna selection schemes as illustrated in Figure 2.4.

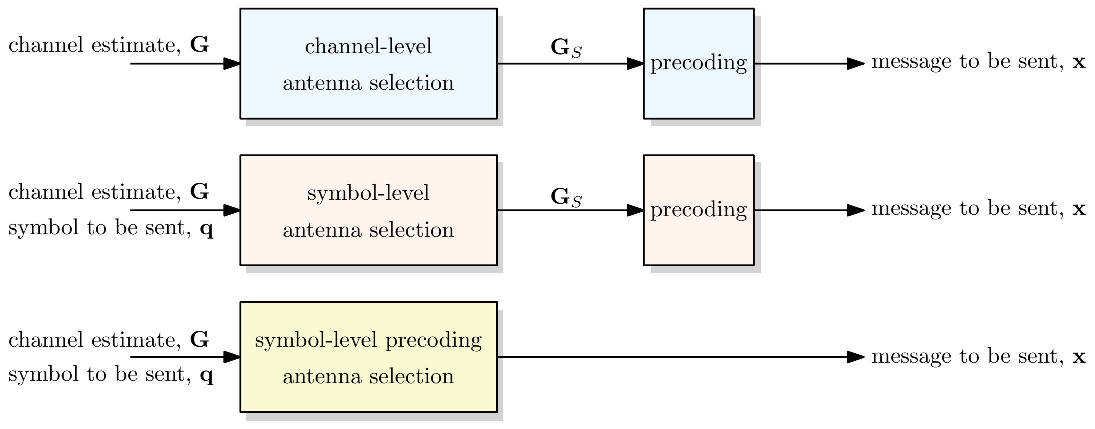

channel estimate, $\mathbf{G}$
channel-level antenna selection
$\mathbf{G}_S$
precoding
message to be sent, $\mathbf{x}$
channel estimate, $\mathbf{G}$
symbol to be sent, $\mathbf{q}$
symbol-level antenna selection
$\mathbf{G}_S$
precoding
message to be sent, $\mathbf{x}$
channel estimate, $\mathbf{G}$
symbol to be sent, $\mathbf{q}$
symbol-level precoding antenna selection
message to be sent, $\mathbf{x}$

Figure 2.4: Illustration of the considered antenna selection schemes.

2.3.4.1 Channel–level antenna selection

By replacing $\mathbf{a}$ with the selection vector $\mathbf{z}$ in equation (2.14) and changing the constraint accordingly, we obtain the resulting optimization problem

$$\begin{aligned} \underset{\mathbf{z}\in\{0,1\}^M}{\text{minimize}} \quad & \|\mathbf{D}\mathbf{z}-\mathbf{b}\|_2^2 \\ \text{subject to} \quad & \|\mathbf{z}\|_0 = S, \end{aligned} \tag{2.34}$$

which, for a given choice of $\mathbf{D}$ and $\mathbf{b}$, the solution fits in the antenna selection approach that only provides the subset of selected antennas. Antenna selection methods operating at the channel–level obtain the subset of selected antennas whenever a new channel estimate is available. Hence, both the target vector $\mathbf{b}$ and dictionary matrix $\mathbf{D}$ are dependent only on the current channel estimate, that is, $\mathbf{b}$ and $\mathbf{D}$ are computed using the channel matrix $\mathbf{G}$. After solving the problem in (2.34), the selection vector $\mathbf{z}$ is used to obtain the S-selected channel matrix $\mathbf{G}_S = \text{diag}(\mathbf{z})\mathbf{G}$. As shown in Figure 2.4, only the channel matrix $\mathbf{G}$ is used as input for channel–level antenna selectors, and then a precoding technique is utilized to obtain the message $\mathbf{x}_S$, $\|\mathbf{x}_S\|_0 = S$. The proposed choice of $\mathbf{D}$ and $\mathbf{b}$ can be found in Algorithm 1.

2.3.4.2 Symbol–level antenna selection

The antenna selection approach operates at the symbol-level whenever a new symbol is available for transmission. In this case, the target vector $\mathbf{b}$ and the dictionary matrix $\mathbf{D}$ in equation (2.34) are the current symbol vector $\mathbf{q}$ and the channel estimate $\mathbf{G}^{\mathrm{T}}$, respectively. After solving the problem in 2.34, the selection vector $\mathbf{z}$ is used to obtain the S-selected channel matrix $\mathbf{G}_S = \mathrm{diag}(\mathbf{z})\mathbf{G}$. Then, $\mathbf{G}_S$ is used by the precoding method to finally obtain the message to be transmitted $\mathbf{x}$. As depicted in Figure 2.4, the S-selected channel matrix is computed considering both the channel estimate $\mathbf{G}$ and the symbol to be transmitted $\mathbf{q}$.

2.3.4.3 Symbol–level precoding antenna selection

By setting $\mathbf{a} = \mathbf{x}$ in equation (2.14) we end up with the optimization problem

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{C}^{M\times 1}}{\text{minimize}} \quad & \|\mathbf{D}\mathbf{x} - \mathbf{b}\|_2^2 \\ \text{subject to} \quad & \|\mathbf{x}\|_0 = S, \end{aligned} \tag{2.35}$$

in which both precoding and antenna selection are performed since the values of $\mathbf{x}$ are not constrained to be zero and one entries. Figure 2.4 demonstrates how the antenna selector directly obtains the message $\mathbf{x}$ to be broadcast.

2.4 Channel–level antenna selection

In this section, we discuss one channel–level antenna selection algorithm inspired by the MP technique. We are interested in the indices of the non-zero entries of the selected vector $\mathbf{z}$ in equation (2.34), which are equivalent to the indices of the chosen codewords in the matching pursuit approach. The following algorithm is named zero forcing greedy antenna selection (ZF-GAS), which was first proposed in [55] with some minor differences.

2.4.1 Zero forcing greedy antenna selection (ZF-GAS)

To apply the MP algorithm for solving the channel–level antenna selection in 2.34, we need to obtain the corresponding target vector $\mathbf{b}$ and dictionary matrix $\mathbf{D}$ based on the current channel estimate. We then seek an expression that captures how alike two matrices are in the sense of providing analogous reception behavior. Therefore, we start by observing the signal received at a certain terminal. When all the BS antennas are active, the received signal at terminal k is

$$y_k = \mathbf{g}_k^{\mathrm{T}}\mathbf{x} + w_k = \mathbf{g}_k^{\mathrm{T}}\mathbf{P}\mathrm{diag}\left(P_T\boldsymbol{\eta}\right)^{1/2}\mathbf{q} + w_k, \tag{2.36}$$

where $\mathbf{P}$ is the precoding matrix, and vector $\boldsymbol{\eta}$ contains the terminal power allocation. In contrast, when only S antennas are active, the terminal k receives

$$y'_k = {\mathbf{g}_S}_k^{\mathrm{T}} \mathbf{x}_S + w_k = \mathbf{g}_k^{\mathrm{T}} \mathrm{diag}\left(\mathbf{z}\right) \mathbf{P}_S \mathrm{diag}\left(P_T \boldsymbol{\eta}\right)^{1/2} \mathbf{q} + w_k, \tag{2.37}$$

where $\mathbf{P}_S$ is the precoding matrix for S active antennas, and $\mathbf{g}_{S_k}$ is the k-th column of matrix $\mathbf{G}_S$ in equation (2.11). For ZF precoding, if $\mathbf{P}_S$ is alike to $\mathbf{P}$[1], then $y_k \simeq y'_k$, which means that the obtained vector $\mathbf{z}$ leads to the minimum reception absolute error, $\|y_k - y'_k\|_2$. For ZF precoding, the sum rate is defined as in [43]

$$\sum_{k=1}^{K} \log_2 \left(1 + \eta_k \rho_{\mathrm{dl}}\right), \tag{2.38}$$

where

$$\rho_{dl} = 10 \, \log_{10} \left(\frac{P_T}{\sigma_w^2} \right) \tag{2.39}$$

is the downlink SNR. Equation (2.38) can be rewritten as

$$\sum_{k=1}^{K} \log_2 \left(1 + \eta_k |\mathbf{g}_k^{\mathrm{T}} \mathbf{p}_k|^2\right), \tag{2.40}$$

and can also be maximized by making $\mathbf{P}_S \simeq \mathbf{P}$ (the contribution of the reduced precoding matrix $\mathbf{P}_S$ is the closest one to the full precoding matrix $\mathbf{P}$ contribution in terms of leading to analogous reception results, following a similarity criterion). This can be verified by noting that $\mathbf{g}_k^{\mathrm{T}} \mathbf{p}_k$ is the k-th diagonal element of $\mathbf{G}^{\mathrm{T}} \mathbf{P}_S$, and $\mathbf{g}_k^{\mathrm{T}} \mathbf{p}_k$ is maximized if $\mathbf{P}_S = \mathbf{P}$ since $\mathbf{G}^{\mathrm{T}} \mathbf{P} = \mathbf{I}_K$. For the ZF precoding, whose solution is $\mathbf{P} = \mathbf{P}_{\mathrm{ZF}} = \mathbf{G}^* (\mathbf{G}^{\mathrm{T}} \mathbf{G}^*)^{-1}$, the approximation $\mathbf{P}_S \simeq \mathbf{P}$ can be rewritten as

$$\mathrm{diag}\left(\mathbf{z}\right) \mathbf{G}^* (\mathbf{G}^{\mathrm{T}} \mathrm{diag}\left(\mathbf{z}\right) \mathbf{G}^*)^{-1} \simeq \mathbf{G}^* (\mathbf{G}^{\mathrm{T}} \mathbf{G}^*)^{-1}. \tag{2.41}$$

Zero-forcing (ZF) precoding is a simple method that decouples the multiuser channel into multiple independent sub-channels and reduces the design to a power allocation problem. Indeed, ZF takes the inter-user interference into account, but neglects the effect of noise. Thus, it performs very well in the high SNR regime, but it is not as efficient under noise-limited scenarios [49].

Instead of finding the optimal $\mathbf{z}$ for equation (2.41), we propose computing sub-optimal $\mathbf{z}$ for

$$(\mathbf{G}^{\mathrm{T}} \mathrm{diag}\left(\mathbf{z}\right) \mathbf{G}^*)^{-1} \simeq (\mathbf{G}^{\mathrm{T}} \mathbf{G}^*)^{-1}, \tag{2.42}$$

using a greedy strategy. Since $\mathbf{P}_S^{\mathrm{H}} \mathbf{P}_S = (\mathbf{G}^{\mathrm{T}} \mathrm{diag}\left(\mathbf{z}\right) \mathbf{G}^*)^{-1}$ and $\mathbf{P}^{\mathrm{H}} \mathbf{P} = (\mathbf{G}^{\mathrm{T}} \mathbf{G}^*)^{-1}$,

[1]We consider that two matrices are alike if the distance between them is short. For example, the distance can be measured by the Frobenius norm of the difference between the matrices [56].

the approximation in equation (2.42) can also be obtained by

$$\mathbf{P}_S^{\mathrm{H}}\mathbf{P}_S \simeq \mathbf{P}^{\mathrm{H}}\mathbf{P}, \tag{2.43}$$

in which $(\cdot)^{\mathrm{H}}$ is the conjugate transpose of $(\cdot)$.

It is convenient to write

$$\mathbf{P}^{\mathrm{H}} = \begin{bmatrix}\tilde{\mathbf{p}}_1 & \tilde{\mathbf{p}}_2 \ldots \tilde{\mathbf{p}}_M\end{bmatrix}, \tag{2.44}$$

where $\mathbf{P}^{\mathrm{H}} \in \mathbb{C}^{K\times M}$ and hence $\tilde{\mathbf{p}}_m \in \mathbb{C}^{K\times 1}$, so that matrix $\mathbf{P}^{\mathrm{H}}\mathbf{P} \in \mathbb{C}^{K\times K}$ can be understood as the summation of rank-one matrices,

$$\mathbf{P}^{\mathrm{H}}\mathbf{P} = \sum_{m=1}^{M} \tilde{\mathbf{p}}_m\tilde{\mathbf{p}}_m^{\mathrm{H}}. \tag{2.45}$$

We can define $\widehat{\mathbf{P}}_S^{\mathrm{H}} = \mathbf{P}^{\mathrm{H}}\mathrm{diag}(\mathbf{z})$ so that

$$\begin{gathered}\widehat{\mathbf{P}}_S^{\mathrm{H}}\widehat{\mathbf{P}}_S = \sum_{m=1}^{M} z_m\tilde{\mathbf{p}}_m\tilde{\mathbf{p}}_m^{\mathrm{H}} = \\ \begin{bmatrix}\tilde{\mathbf{p}}_1\tilde{\mathbf{p}}_1^{\mathrm{H}} & \tilde{\mathbf{p}}_2\tilde{\mathbf{p}}_2^{\mathrm{H}} \cdots \tilde{\mathbf{p}}_M\tilde{\mathbf{p}}_M^{\mathrm{H}}\end{bmatrix}\begin{bmatrix} z_1\mathbf{I}_K \\ z_2\mathbf{I}_K \\ \vdots \\ z_m\mathbf{I}_K\end{bmatrix} \simeq \sum_{m=1}^{M}\tilde{\mathbf{p}}_m\tilde{\mathbf{p}}_m^{\mathrm{H}}\end{gathered} \tag{2.46}$$

is a surrogate to equation (2.43), which is more convenient for MP purposes.

Although the approximation in equation (2.46) is between two matrices and not between two vectors, we can get around this issue by the following manipulations. Consider the auxiliary block matrix

$$\mathbf{D}_{\mathrm{aux}} = \begin{bmatrix}\mathbf{C}_1 & \mathbf{C}_2 \ldots \mathbf{C}_M\end{bmatrix} \tag{2.47}$$

in which $\mathbf{C}_m = \tilde{\mathbf{p}}_m\tilde{\mathbf{p}}_m^{\mathrm{H}}$, for $m \in \{1, \cdots, M\}$. Also, let $\mathbf{R}_1 = \mathbf{P}^{\mathrm{H}}\mathbf{P}$ so that we can try to find the best match for $\mathbf{R}_1$ in $\mathbf{D}_{\mathrm{aux}}$ by computing

$$\langle \mathbf{R}_1, \mathbf{C}_m\rangle = \mathrm{tr}(\mathbf{R}_1^{\mathrm{H}}\mathbf{C}_m) \tag{2.48}$$

for $m \in \{1, \cdots, M\}$. Since $\mathbf{R}_1 \in \mathbb{C}^{K\times K}$ and $\mathbf{C}_m \in \mathbb{C}^{K\times K}$, the trace of the product can be rewritten as

$$\mathrm{tr}(\mathbf{R}_1^{\mathrm{H}}\mathbf{C}_m) = \sum_{i=1}^{K}\left(\mathbf{R}_1^{\mathrm{H}}\mathbf{C}_m\right)_{i,i} = \sum_{i=1}^{K}\sum_{j=1}^{K}(\mathbf{R}_1^{\mathrm{H}})_{i,j}(\mathbf{C}_m)_{j,i}, \tag{2.49}$$

in which $(\cdot)_{i,j}$ is the element of a matrix placed at the i-th row and j-th column. The last summation in equation (2.49) is the sum of entry-wise products of elements and thus can be rewritten as the vector inner product

$$\sum_{i=1}^{K}\sum_{j=1}^{K}(\mathbf{R}_1^{\mathrm{H}})_{i,j}(\mathbf{C}_m)_{j,i} = \mathrm{vec}(\mathbf{R}_1)^{\mathrm{H}}\,\mathrm{vec}(\mathbf{C}_m) = \mathbf{r}_1^{\mathrm{H}}\mathbf{c}_m, \tag{2.50}$$

using the vectorization operator: the vec$(\cdot)$ operator takes a matrix as input and outputs a column vector which is obtained by transposing the rows of the input matrix and stacking them. Hence, by using $\mathbf{b} = \mathrm{vec}(\mathbf{P}^{\mathrm{H}}\mathbf{P})$ as target vector, and

$$\mathbf{D}_{\mathrm{ZF}} = \left[\mathrm{vec}(\tilde{\mathbf{p}}_1\tilde{\mathbf{p}}_1^{\mathrm{H}}) \quad \mathrm{vec}(\tilde{\mathbf{p}}_2\tilde{\mathbf{p}}_2^{\mathrm{H}})\ldots\mathrm{vec}(\tilde{\mathbf{p}}_M\tilde{\mathbf{p}}_M^{\mathrm{H}})\right] \tag{2.51}$$

as dictionary matrix, we obtain the zero forcing greedy antenna selection (ZF-GAS) detailed in Algorithm 1. This name comes from the fact that the zero-forcing scheme is considered in equation (2.41), and the active antennas are chosen in a greedy fashion inspired by matching pursuit. The columns of $\mathbf{D}_{\mathrm{ZF}}$ represent the codewords, for example $\mathbf{c}_1 = \mathrm{vec}(\mathbf{C}_1) = \mathrm{vec}(\tilde{\mathbf{p}}_1\tilde{\mathbf{p}}_1^{\mathrm{H}})$.

Algorithm 1 : Zero forcing greedy antenna selection (ZF-GAS)

```
1) Input: $\mathbf{b} = \mathrm{vec}(\mathbf{P}^{\mathrm{H}}\mathbf{P})$, $\mathcal{I} = \{1,\ldots,M\}$, $0 < \gamma \leq 1$,
          $\mathbf{D}_{\mathrm{ZF}} = [\mathbf{c}_1\ldots\mathbf{c}_M]$, where $\mathbf{c}_m = \mathrm{vec}(\tilde{\mathbf{p}}_k\tilde{\mathbf{p}}_k^{\mathrm{H}})$ and $\tilde{\mathbf{p}}_k$ is the $k$-th column of
          matrix $\mathbf{P}^{\mathrm{H}} = (\mathbf{G}^*(\mathbf{G}^{\mathrm{T}}\mathbf{G}^*)^{-1})^{\mathrm{H}}$.
2) Initialization: $i = 1$, $\mathbf{r}_1 = \mathbf{b}$, $\mathbf{z} = \mathbf{0}_M$.
3) Repeat until $i = S$:
   a) Find the closest codeword, i.e., find $m_i \in \mathcal{I}$ such that
      $\langle\mathbf{r}_i,\mathbf{c}_{m_i}\rangle = \max_{m_i\in\mathcal{I}}\ \{|\langle\mathbf{r}_i,\mathbf{c}_{mi}\rangle|\}$.
   b) Choose $z_{m_i} = 1$.
   c) Set $f_i = \gamma\langle\mathbf{r}_i,\mathbf{c}_{m_i}\rangle$
   d) Let $\mathbf{r}_{i+1} = \mathbf{r}_i - f_i\mathbf{c}_{m_i}$.
   e) $\mathcal{I} = \mathcal{I} - \{m_i\}$.
   f) Increment $i$.
4) Compute the $S$-selected channel matrix $\mathbf{G}_S = \mathrm{diag}\,(\mathbf{z})\,\mathbf{G}$.
5) Output: $\mathbf{G}_S$.
```

With the output of algorithm 1, a precoding scheme can be applied, and thus a precoded message $\mathbf{x}_S$ is produced. For linear precoding, the vector to be transmitted can be computed as

$$\mathbf{x}_S = \mathbf{P}_S\mathrm{diag}\,(\boldsymbol{\eta})^{1/2}\,\mathbf{q}. \tag{2.52}$$

Moreover, the transmitted signal can be scaled by a desired average transmission power P_T, yielding

$$\tilde{\mathbf{x}}_S = \frac{\sqrt{P_T}}{\sqrt{E[|\mathbf{x}_S|^2]}}\mathbf{x}_S, \tag{2.53}$$

where $E[|\mathbf{x}_S|^2]$ is the expected value of $|\mathbf{x}_S|^2$.

2.5 Symbol–level antenna selection

In the downlink of a massive MIMO system, the BS aims to transmit vectors of the form $\mathbf{q} \in \mathbb{C}^{(K\times 1)}$ containing the symbols intended to each terminal $k \in \{1, \cdots, K\}$. In fact, the BS generates a block of L message vectors $[\mathbf{q}_1 \dots \mathbf{q}_L]$. First, the BS performs precoding so that the transmitted signal is $\mathbf{x} = \mathbf{P}\mathrm{diag}\left(\boldsymbol{\eta}\right)^{1/2}\mathbf{q}$ when all the transmit antennas are active. The precoding matrix $\mathbf{P}$ can be the same for all $\mathbf{q}_l$ in the block as for the ZF-GAS algorithm, in which one computes matrix $\mathbf{P}$ only once. In contrast, matrix $\mathbf{P}$ can be computed for every $\mathbf{q}_l$ in the block, giving rise to a symbol–level precoding.

In this section, we propose one algorithm for symbol–level antenna selection. For this symbol–level method, a precoding technique must be applied after selection. This algorithm was first introduced in [55] with some minor differences as the dictionary normalization and the residue rate of decay.

2.5.1 Zero forcing symbol–level matching pursuit antenna selection (ZFSL-MPAS)

Consider the message transmitted by the M BS antennas

$$\mathbf{P}\mathrm{diag}\left(P_T\boldsymbol{\eta}\right)^{1/2}\mathbf{q} = \mathbf{x} \tag{2.54}$$

and by left-multiplying both sides of equation (2.54) by $\mathbf{G}^{\mathrm{T}}$, and replacing matrix $\mathbf{P}$ by the zero-forcing precoding matrix $\mathbf{P}_{\mathrm{ZF}} = \mathbf{G}^*(\mathbf{G}^{\mathrm{T}}\mathbf{G}^*)^{-1}$, we obtain

$$\mathrm{diag}\left(P_T\boldsymbol{\eta}\right)^{1/2}\mathbf{q} = \mathbf{G}^{\mathrm{T}}\mathbf{x}. \tag{2.55}$$

Fortunately, we can bring it to the MP point of view as: $\mathbf{b} = \mathrm{diag}\left(\boldsymbol{\eta}\right)^{1/2}\mathbf{q}$ and $\mathbf{D} = \mathbf{G}^{\mathrm{T}}$ and consider $\mathbf{x}$ as the selection vector $\mathbf{z}$. That is, we need to build $\mathbf{z}$ that reflects the most informative elements of $\mathbf{x}$. The resulting problem can be formulated as

$$\begin{aligned} \underset{\mathbf{z}\in\{0,1\}^M}{\text{minimize}} \quad & \left\|\mathbf{G}^{\mathrm{T}}\mathbf{z} - \mathrm{diag}\left(\boldsymbol{\eta}\right)^{1/2}\mathbf{q}\right\|_2^2 \\ \text{subject to} \quad & \|\mathbf{z}\|_0 = S. \end{aligned} \tag{2.56}$$

Since P_T is fixed, we only need to inform the portions η_k in the problem (2.56). Therefore, we obtain Algorithm 2, named zero forcing symbol–level matching pursuit antenna selection (ZFSL-MPAS). This name comes from the ZF precoding scheme in equation (2.55); the resulting algorithm is a symbol–level, and a greedy method is employed to choose the active antennas. The vector to be transmitted $\mathbf{x}_S$ can be computed as in equation (2.52) and scaled as in equation (2.53) to account for the transmission power employed.

Algorithm 2 : Zero forcing symbol–level matching pursuit antenna selection (ZFSL-MPAS)

```
1) Input: $\mathbf{b} = \mathrm{diag}(\boldsymbol{\eta})^{1/2}\mathbf{q}$, $\mathcal{I} = \{1, \ldots, M\}$, $0 < \gamma \leq 1$,
          $\mathbf{D} = [\mathbf{c}_1 \ldots \mathbf{c}_M]$, where $\mathbf{c}_k$ is the $k$-th column of matrix $\mathbf{G}^{\mathrm{T}}$.
2) Initialization: $i = 1$, $\mathbf{r}_1 = \mathbf{b}$, $\mathbf{z} = \mathbf{0}_M$.
3) Repeat until $i = S$:
    a) Find the closest codeword, i.e., find $m_i \in \mathcal{I}$ such that
       $\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle = \max_{m_i \in \mathcal{I}} \{|\langle \mathbf{r}_i, \mathbf{c}_{mi} \rangle|\}$.
    b) Choose $z_{m_i} = 1$.
    c) Set $f_i = \gamma \langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle$.
    d) Let $\mathbf{r}_{i+1} = \mathbf{r}_i - f_i \mathbf{c}_{m_i}$.
    e) $\mathcal{I} = \mathcal{I} - \{m_i\}$.
    f) Increment $i$.
4) Compute the $S$-selected channel matrix $\mathbf{G}_S = \mathrm{diag}(\mathbf{z})\,\mathbf{G}$.
5) Output: $\mathbf{G}_S$.
```

2.6 Symbol–level precoding antenna selection

Essentially, the matching pursuit technique relies on representing a target vector by using a combination of the codewords of a redundant dictionary that best matches the target vector. If the target vector is the scaled vector of symbols $\mathrm{diag}(\boldsymbol{\eta})^{1/2}\mathbf{q}$ and the dictionary matrix is the transposed channel matrix $\mathbf{G}^{\mathrm{T}}$, the resulting sparse vector that contains the inner products could be an approximation of the message to be transmitted $\mathbf{x}$. Consequently, another symbol–level antenna selector can be obtained in which message $\mathbf{x}$ is directly obtained.

In this section, we propose two symbol–level algorithms that perform both antenna selection and precoding.

2.6.1 Symbol–level matching pursuit precoding antenna selection (SL-MPPAS)

Similarly to the approach in the ZFSL-MPAS, ZF precoding is aimed to solve the optimization problem

$$\begin{aligned} \underset{\mathbf{x} \in \mathbb{C}^{M \times 1}}{\text{minimize}} \quad & \left\| \mathbf{G}^T \mathbf{x} - \mathrm{diag}(\boldsymbol{\eta})^{1/2}\mathbf{q} \right\|_2^2 \\ \text{subject to} \quad & \|\mathbf{x}\|_0 = S, \end{aligned} \tag{2.57}$$

without the constraint on the values of $\mathbf{x}$. In equation (2.57), vector $\mathbf{x}$ is a precoded version of the symbol vector $\mathbf{q}$ using only S active antennas. Hence, the sparse vector in the MP problem is the message we desire to transmit. The resulting algorithm is called symbol-level matching pursuit precoding antenna selection (SL-MPPAS) which is described in Algorithm 3. This name comes from the fact that the resulting algorithm is a symbol–level, and it not only chooses the active antennas but also produces the precoded signal $\mathbf{x}$ using the matching pursuit technique. Observe that the non-zero elements of vector $\mathbf{x}$ are the inner products between the selected codewords and the residues. The vector to be transmitted $\mathbf{x}_S$ is the output of Algorithm 3, but $\mathbf{x}_S$ can be scaled as in equation (2.53) to account for the transmission power employed.

2.6.2 Symbol–level matching pursuit with generalized bit planes precoding antenna selection (SL-MPGBPPAS)

Alternatively, we can impose that the non-zero elements of $\mathbf{x}$ are quantized by using the MPGBP approach, as described in Algorithm 4, named symbol–level matching pursuit with generalized bit planes precoding antenna selection (SL-MPGBPPAS). This name comes from the fact that the resulting algorithm is a symbol–level, and it not only chooses the active antennas but also produces the precoded signal $\mathbf{x}$ using the matching pursuit generalized bit planes technique. Here, the non-zero elements of vector $\mathbf{x}$ are the quantized inner products between the selected codewords and the residues. The vector to be transmitted $\mathbf{x}_S$ is the output of Algorithm 4, but $\mathbf{x}_S$ can be scaled as in equation (2.53) to determine the transmission power employed.

Algorithm 3 : Symbol–level matching pursuit precoding antenna selection (SL-MPPAS)

```
1) Input: $\mathbf{b} = \mathrm{diag}(\boldsymbol{\eta})^{1/2}\,\mathbf{q}$ , $\mathcal{I} = \{1, \dots, M\}$, $0 < \gamma \leq 1$,
          $\mathbf{D} = [\mathbf{c}_1 \dots \mathbf{c}_M]$, where $\mathbf{c}_k$ is the $k$-th column of matrix $\mathbf{G}^{\mathrm{T}}$.
2) Initialization: $i = 1$, $\mathbf{r}_1 = \mathbf{b}$, $\mathbf{x} = \mathbf{0}_M$.
3) Repeat until $i = S$:
    a) Find the closest codeword, i.e., find $m_i \in \mathcal{I}$ such that
       $\langle \mathbf{r}_i,\mathbf{c}_{m_i} \rangle = \max_{m_i \in \mathcal{I}} \{|\langle \mathbf{r}_i,\mathbf{c}_{mi} \rangle|\}$.
    b) Set $f_i = \gamma \langle \mathbf{r}_i,\mathbf{c}_{m_i} \rangle$.
    c) $x_{m_i} = f_i$.
    d) Let $\mathbf{r}_{i+1} = \mathbf{r}_i - f_i \mathbf{c}_{m_i}$.
    e) $\mathcal{I} = \mathcal{I} - \{m_i\}$.
    f) Increment $i$.
4) Output: $\mathbf{x}$.
```

The main advantage of the precoder antenna selectors algorithms, here named SL-MPPAS and SL-MPGBPPAS, is that their outputs are already the message to be

Algorithm 4 : Symbol–level matching pursuit with generalized bit planes precoding antenna selection (SL-MPGBPPAS)

```
1) Input: $\mathbf{b} = \text{diag}(\boldsymbol{\eta})^{1/2}\,\mathbf{q}$, $\mathcal{I} = \{1, \ldots, M\}$, $0 < \gamma \leq 1$,
          $\mathbf{D} = [\mathbf{c}_1 \ldots \mathbf{c}_M]$, where $\mathbf{c}_k$ is the $k$-th column of matrix $\mathbf{G}^{\mathrm{T}}$.
2) Initialization: $i = 1$, $\mathbf{r}_1 = \mathbf{b}$, $\mathbf{x} = \mathbf{0}_M$.
3) Repeat until $i = S$:
    a) Find the closest codeword, i.e., find $m_i \in \mathcal{I}$ such that
       $\langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle = \max_{m_i \in \mathcal{I}} \{|\langle \mathbf{r}_i, \mathbf{c}_{mi} \rangle|\}$.
    b) Set $f_i = \gamma \langle \mathbf{r}_i, \mathbf{c}_{m_i} \rangle$.
    c) Choose $\mu_{i_1} = \lceil \log_2 \left( \frac{3}{\text{Re}\{f_i\}} \right) \rceil$ and $\mu_{i_2} = \lceil \log_2 \left( \frac{3}{\text{Im}\{f_i\}} \right) \rceil$.
    d) Let $\alpha^{\mu_i} = \frac{\text{Re}\{f_i\}}{||\text{Re}\{f_i\}||} (\frac{1}{2})^{\text{Re}\{\mu_{i_1}\}} + \frac{\text{Im}\{f_i\}}{||\text{Im}\{f_i\}||} j (\frac{1}{2})^{\text{Re}\{\mu_{i_2}\}}$.
    e) $x_{m_i} = \alpha^{\mu_i}$.
    f) Let $\mathbf{r}_{i+1} = \mathbf{r}_i - \alpha^{\mu_i} \mathbf{c}_{m_i}$.
    g) $\mathcal{I} = \mathcal{I} - \{m_i\}$.
    h) Increment $i$.
4) Output: $\mathbf{x}$.
```

transmitted $\mathbf{x}$ using a subset of S active antennas, which reduces the complexities of the algorithms. The previous algorithms, which are only antenna selectors, output the reduced channel matrix $\mathbf{G}_S$ and need to perform an external precoding scheme to generate the message to be transmitted. Since the SL-MPGBPPAS algorithm provides a message formed by quantized elements, the requirements on the amplifier linearity at the BS is reduced, further reducing the BS cost.

2.7 Computational complexity

In this section, we quantify the complexity of the presented algorithms by counting the required number of flops to compute a block with L message vectors $\mathbf{x}$. We consider that during the coherence time τ, i.e., the time in which the impulse response of the channel can be considered invariant, we can transmit an L-symbol transmission block. A flop is a real floating-point operation [57]. Real addition and subtraction are counted as one flop. Likewise, real multiplication, division, and exponential are also counted as one flop. Therefore, we consider all real operations as one flop, including division and exponential. On the other hand, complex addition and multiplication have two and six flops, respectively [58]. The complexity (in flops) of each proposed antenna selection algorithm is summarized in Table 2.1.

The initial residue used in ZF-GAS is $\mathbf{b} = \text{vec}((\mathbf{G}^{\mathrm{T}}\mathbf{G}^*)^{-1})$ and requires $28K^3/3 - 3K^2 + 8M^2K$ flops, considering that the QR decomposition was employed to perform the matrix inversion operation. The initial residue $\mathbf{b} = \mathbf{q}$ employed in ZFSL-MPAS, SL-MPPAS and SL-MPGBPPAS requires zero flops.

Table 2.1: Number of flops required to compute a block of L message vectors by the proposed antenna selection algorithms; M is the number of BS antennas, S is the number of selected antennas, K is the number of terminals and L is the length of the transmitted block

Algorithm	**Initial residue and dictionary**	**Main Loop**	**Block of L messages**	**Total number of flops**
ZF-GAS	$\frac{28}{3}K^3 - 3K^2 + 8M^2K + 6MK^2$	$4 + 2SK^2 + (2MS - (S^2 - S))4K^2$	**ZF**: $[\frac{28}{3}K^3 + 8SK^2 - 3K^2 + 8MK + 8MK^2]L$ **MR**: $[8MK + 8MK^2]L$	**ZF**: $\frac{28}{3}K^3(1+L) + 8M^2K + 8MSK^2 + MK^2(6+8L) + SK^2(6+8L) - 4S^2K^2 - 3K^2(1+L) + 8MKL + 5$ **MR**: $\frac{28}{3}K^3 + 8M^2K - 3K^2 + 8MSK^2 + MK^2(6+8L) + 6SK^2 + 8MKL + 4$
ZFSL-MPAS	0	$[4 + 2SK + (2MS - (S^2 - S))4K]L$	**ZF**: $[\frac{28}{3}K^3 + 8SK^2 - 3K^2 + 8MK + 8MK^2]L$ **MR**: $[8MK + 8MK^2]L$	**ZF**:$(\frac{28}{3}K^3 + 8MK^2 + 8MK + 8MSK - 3K^2 + 8SK^2 - 4S^2K + 6SK + 5)L$ **MR**: $(\frac{28}{3}K^3 + 8M^2K + 16MK^2 + 8MSK + 6SK - 3K^2 - 4S^2K + 8MK + 5)L$
SL-MPPAS	0	$[4 + 6SK + 8MSK - 4S^2K]L$	0	$(4 + 6SK + 8MSK - 4S^2K)L$
SL - MPGBPPAS	0	$[4 + 6SK + 8MSK - 4S^2K + 3(S+1)]L$	0	$[4 + 6SK + 8MSK - 4S^2K + 3(S+1)]L$

To generate the dictionary matrix for ZF-GAS, we need to compute M outer-products that require $6MK^2$ flops in total. The ZFSL-MPAS, SL-MPPAS and SL-MPGBPPAS algorithms use $\mathbf{G}^{\mathrm{T}}$ as dictionary matrix which does not require extra computation.

The main loop of ZF-GAS, ZFSL-MPAS, and SL-MPPAS consists of S vector additions and $MS - (S^2 - S)/2$ inner products. It requires $2SK^2 + 4MSK^2 - (S^2 - S)4K^2$ flops in total for ZF-GAS and $6SK + 8MKS - 4KS^2$ for ZFSL-MPAS and SL-MPPAS. The main loop of SL-MPGBPPAS is almost the same as the SL-MPPAS. The difference is that the SL-MPGBPPAS requires $(S+1)$ additional multiplications and hence $3(S+1)$ additional flops.

Moreover, ZF-GAS, ZFSL-MPAS, SL-MPPAS, and SL-MPGBPPAS algorithms compute the sub-optimal factor $0 < \gamma \leq 1$ once. For $\gamma = \frac{K}{S}\left(\frac{K}{K+20}\right)$, it is required one real addition and three real multiplications, amounting to four flops.

The last stage consists of computing the S-selected message vector $\mathbf{x}_S$. Algorithms ZF-GAS and ZFSL-MPAS are only antenna selectors and thus require a precoding scheme to obtain the message vector $\mathbf{x}_S = \mathbf{P}_S \mathrm{diag}\left(\boldsymbol{\eta}\right)^{1/2}\mathbf{q}$. For ZF precoding, $\mathbf{P}_S = (\mathbf{G}\mathrm{diag}\left(\mathbf{z}\right))^*(\mathbf{G}_S^{\mathrm{T}}\mathbf{G}_S^*)^{-1}$ and then it requires $28K^3/3 - 3K^2 + 8SK^2 + 8MSK + 8SK$ flops to compute $\mathbf{x}$. For MR precoding, $8MSK + 8SK$ flops are required to compute $\mathbf{x}$.

As shown in Table 2.1, the ZF-based methods have almost the same complexity in flops due to the matrix inversion operation. On the other hand, SL-MPPAS and SL-MPGBPPAS are the fastest algorithms since their complexity is concentrated on the main loop where no matrix inversion is needed.

We consider the approximate version of the maximum capacity antenna selection (MCAS) algorithm [3] by using one iteration of the log-barrier method [59]. Using this strategy, the inequality constraints are implicit in the objective function and hence the optimization problem can be solved via Newton Method [40]. Therefore, the complexity of the MCAS algorithm is concentrated on the Newton step computation. The Newton step depends on the gradient vector and the Hessian matrix of the objective function.

The expressions of the gradient and the Hessian matrix are the same as the ones shown in [59], as well as the Newton step. To compute the gradient vector, $M(\frac{28}{3}K^3 + 8M + 6MK^2 + 13K^2 + 16)$ flops are required, since M matrix inversion operations are performed using QR decomposition. The Hessian matrix costs $8M^2K^2 + 6M^2 + 8MK^2 + 11M + 1$ flops, and its inverse requires $\frac{28}{3}M^3 - 3M^2$ additional flops. The total number of flops required to compute the transmission block is shown in Table 2.2, in which U is the number of iterations required to achieve convergence. Typically, $U < 10$, as reported in [59].

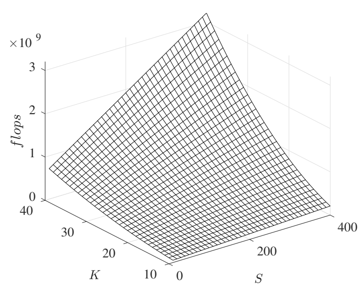
To produce the selection vector $\mathbf{z}$, the power-based antenna selection (PBAS) algorithm [2, 60, 61] requires $8MK^2$ flops since M inner-products are computed, whereas the RAS algorithm [1] requires zero flops since additions and multiplications are not performed. The number of flops required to compute the transmission block using both ZF and MR precoding schemes are also exposed in Table 2.2.

Table 2.2: Number of flops required to compute a block of L message vectors by existing antenna selection algorithms; M is the number of BS antennas, S is the number of selected antennas, K is the number of terminals, L is the length of the transmitted block and U is the number of iterations required to achieve convergence in MCAS

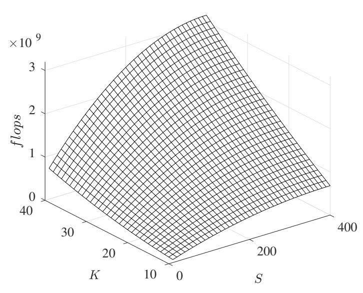
Algorithm	**Main**	**Block of L messages**	**Total number of flops**
MCAS [3]	$UL(\frac{28}{3}M^3 + 43M^2 + \frac{28}{3}K^3M + 14M^2K^2 + 21MK^2 + 35M + 3)$	**ZF**: $(\frac{28}{3}K^3 + 8SK^2 - 3K^2 + 8MK + 8MK^2)L$ **MR**: $(8MK + 8MK^2)L$	**ZF**:$[U(\frac{28}{3}M^3 + 43M^2 + \frac{28}{3}K^3M + 14M^2K^2 + 21MK^2 + 35M + 3) + \frac{28}{3}K^3 + 8SK^2 - 3K^2 + 8MK + 8MK^2]L$ **MR**: $[U(\frac{28}{3}M^3 + 43M^2 + \frac{28}{3}K^3M + 14M^2K^2 + 21MK^2 + 35M + 3) + 8MK + 8MK^2]L$
PBAS [2]	$8LMK^2$	**ZF**: $(\frac{28}{3}K^3 + 8SK^2 - 3K^2 + 8MK + 8MK^2)L$ **MR**: $(8MK + 8MK^2)L$	**ZF**:$(\frac{28}{3}K^3 + 8SK^2 - 3K^2 + 8MK + 16MK^2)L$ **MR**: $(8MK + 16MK^2)L$
RAS [1]	0	**ZF**: $(\frac{28}{3}K^3 + 8SK^2 - 3K^2 + 8MK + 8MK^2)L$ **MR**: $(8MK + 8MK^2)L$	**ZF**:$(\frac{28}{3}K^3 + 8SK^2 - 3K^2 + 8MK + 8MK^2)L$ **MR**: $(8MK + 8MK^2)L$

The algorithm complexities are illustrated in Figures 2.5 and 2.6 for a fixed number of BS antennas $M = 400$. Since the complexity of SL-MPPAS is almost the same as the SL-MPGBPPAS complexity, we only show it in Figures 2.5 and 2.6 since it has the highest complexity.

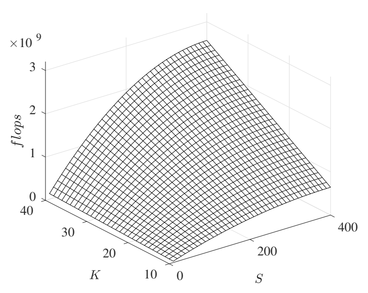
In Figure 2.5, we consider a long coherence time; hence, the transmission block size is $L = 100$, whereas, in Figure 2.6, $L = 25$ represents a short coherence time. Even with only one iteration $U = 1$ considered, MCAS is the most complex among the tested algorithms. When we have a long coherence time, i.e., $L = 100$, channel–level antenna selection is preferred. In contrast, when we have a short coherence time, i.e., $L = 25$, symbol–level antenna selection is a good choice. Moreover, a symbol–level like SL-MPPAS has almost the same complexity as channel–level



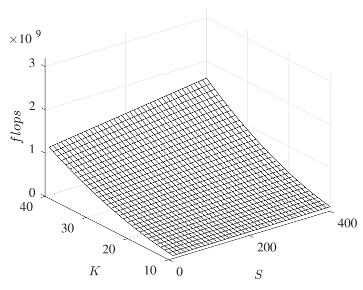
(a) ZF-GAS (Algorithm 1)



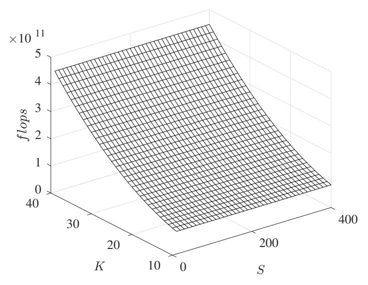
(b) ZFSL-MPAS (Algorithm 2)



(c) SL-MPPAS (Algorithm 3)



(d) PBAS [2]



(e) MCAS [3] with $U = 1$

Figure 2.5: Number of flops as a function of S and K. In this case, $M = 400$, $L = 100$.

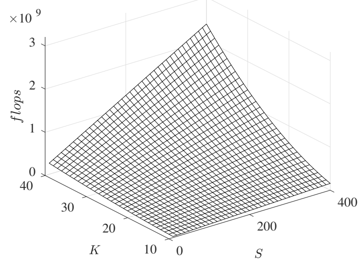
PBAS, with the advantage of achieving better performances in terms of bit error rate (BER) and mean squared error (MSE), as will be seen in the simulations.

2.8 Simulation results

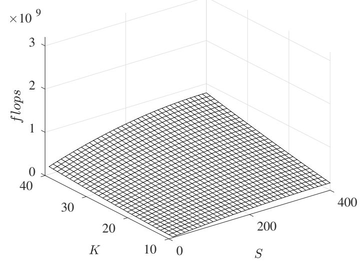
In this section, we present simulation results to verify the performance of the proposed antenna selection approaches. Our experiments are carried out in a single-cell system where $K = 12$ single-antenna terminals are served by a BS equipped with $M = 400$ antennas. On average, $M \geq 100$ BS antennas is required to observe the channel hardening effect.

As in [3], we assume equal power allocation among the terminals when choosing the subset of active antennas. The power allocation can be treated as a side problem that can be solved using water-filling algorithms [40, 62] or an approximation [43]. We consider both rich and poor scattering environments where the downlink channels are modeled using uncorrelated and correlated Rayleigh fading. To simplify, we consider the large-scale coefficient $\beta_k = 1$.

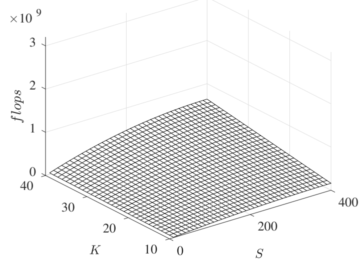
In the uplink phase, BPSK pilot symbols are transmitted by the terminals and used to estimate the channel matrix using minimum mean square error (MMSE) estimator, which assumes the knowledge of the channel noise variance. The pilot matrix $\mathbf{\Phi} \in \mathbb{C}^{\tau_p \times K}$ is generated as a Walsh-Hadamard matrix, in which $\tau_p = K$ is



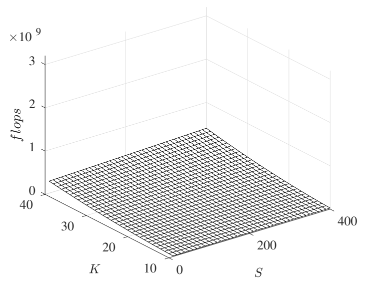
(a) ZF-GAS (Algorithm 1)



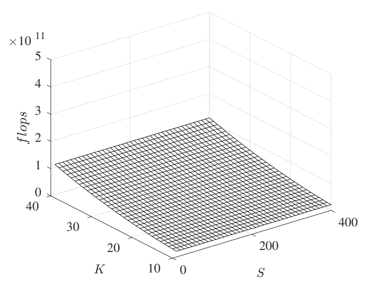
(b) ZFSL-MPAS (Algorithm 2)



(c) SL-MPPAS (Algorithm 3)



(d) PBAS [2]



(e) MCAS [3] with $U = 1$

Figure 2.6: Number of flops as a function of S and K. In this case, $M = 400$, $L = 25$. We use a different scale for MCAS since the order of magnitude of MCAS complexity is two times greater than the other methods.

the pilot duration [48] and hence the complete set of pilot signals transmitted by the terminals is given by $\mathbf{X}_p = \sqrt{\tau_p}\mathbf{\Phi}^{\mathrm{H}}$. The BS can estimate the channel matrix by right-multiplying the received signal $\mathbf{Y}_p = \sqrt{\tau_p}\mathbf{G}\mathbf{\Phi}^{\mathrm{H}} + \mathbf{W}$ by the pilot matrix. The noise represented by $\mathbf{W}$ is generated so that the uplink SNR is $\rho_{ul} = 3$ dB. The MMSE estimator [48] is then applied in the columns $\mathbf{y}'_k$ of the noisy estimate $\mathbf{Y}'_p = \sqrt{\tau_p}\mathbf{G} + \mathbf{W}$, obtaining the estimate of the channel response for each terminal k,

$$\hat{\mathbf{g}}_k = \frac{\sqrt{\tau_p}\beta_k}{1 + \tau_p\beta_k}\mathbf{y}'_k. \tag{2.58}$$

The simulation scenarios with the true channel are referred to as perfect CSI. In this case, no channel estimation is performed, and the only impairment is the noise effect. We have also verified how the performances of the antenna selection methods are affected when CSI is not perfect. In this case, we perform channel estimation, and we refer to such a scenario as partial CSI.

In the downlink phase, the BS generates a block of 50 message vectors $[\mathbf{q}_1 \ldots \mathbf{q}_{50}]$, where each vector $\mathbf{q} \in \mathbb{C}^{K\times 1}$ contains a stream of 4-QAM or QPSK samples. In particular, a stream of 16-QAM samples is used in simulated scenario 5 to verify if the proposed methods support higher-order modulations. The BS applies the antenna selection algorithms listed in Table 2.3, where the proposed antenna selection algorithms are highlighted in boldface. The proposed algorithms utilize γ as in equation (2.33), with $T = 20$. For comparison, we consider the algorithms closely related to our methods. Moreover, we consider in the simulations the case in which all the antennas are active, referred as "Full".

Table 2.3: Antenna selection algorithms evaluated in the BER simulations.

Algorithm	**Source**	**Description**	**Type**
ZF-GAS	Proposed	Zero forcing Greedy Antenna Selection	channel–level
RAS	[1]	Random Antenna Selection	channel–level
MCAS	[3]	Maximum Capacity Antenna Selection	channel–level
PBAS	[2]	Power based Antenna selection	channel–level
ZFSL-MPAS	Proposed	Zero forcing Symbol–Level Greedy Antenna Selection	symbol–level
SL-MPPAS	Proposed	Symbol–Level Matching Pursuit Precoding Antenna Selection	symbol–level
SL-MPGBPPAS	Proposed	Symbol–Level Matching Pursuit with Generalized Bit Planes Precoding Antenna Selection	symbol–level
LASSO	[4, 5]	Least Absolute Shrinkage and Selection Operator	symbol–level

With only S out of M active BS antennas, the signal to be transmitted is produced. Considering an AWGN with fixed variance $\sigma_w^2 = 0.9$, the transmission signal power P_T is varied in order to obtain a downlink SNR $\rho_{dl} \in \{-12{,}12\}$ dB. The downlink SNR is obtained using equation (2.39) and it is repeated as follows for the reader's convenience

$$\rho_{dl} = 10 \log_{10}\left(\frac{P_T}{\sigma_w^2}\right). \tag{2.59}$$

Since the noise level is fixed, comparing the SNR is the same as comparing the transmission power required to deal with a certain level of noise. The received symbols are detected following the element-wise minimum Euclidean distance criterion as in [63]. The received symbols are scaled so that the constellation has unit average power to be fairly compared with the reference constellation.

The averaged BER and MSE are presented as a function of the SNR. We show the simulation results in several distinct scenarios in which the channel model and CSI knowledge are explored. All the simulation results are obtained by averaging 100 Monte Carlo runs. The simulations are performed using MATLAB software, and the convex optimization problems are solved using CVX [64].

In scenario 1, we consider a rich scattering environment where the channel is modeled using uncorrelated Rayleigh fading. We also consider that the BS has perfect CSI knowledge, that is, no channel estimation is performed. Moreover, the ZF precoding scheme is used in the antenna selection methods that need to perform external precoding after selection. A variation of scenario 1 called scenario 2 uses MR as the external precoding rather than ZF. Scenario 3 addresses the case in which the BS has partial CSI knowledge. Scenario 3 also provides the comparison between a channel model for rich and poor scattering environments for $S = 100$ selected antennas. In scenario 4, a comparison between the proposed methods and their pair counterparts based on convex optimization is provided for $S = 40$ selected antennas, considering perfect CSI knowledge at the BS in a rich scattering environment. Scenario 5 is a version of scenario 1 in which 16-QAM symbols are used

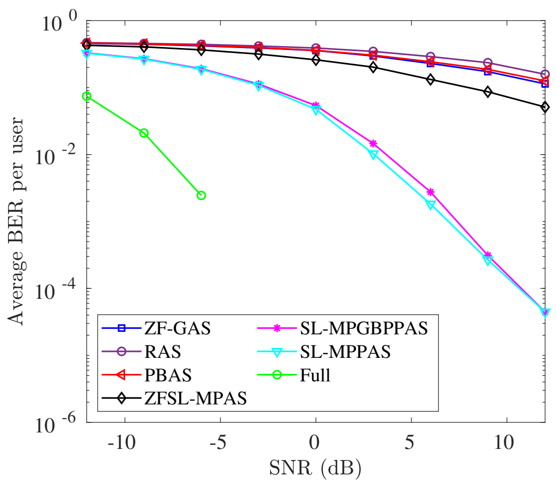
2.8.1 Scenario 1: perfect CSI knowledge, 4-QAM, rich scattering and ZF precoding

Figure 2.7 depicts the average BER per user for scenario 1 when $S = 12, 40, 100$ and 200 and 4-QAM samples are transmitted. For a fixed transmission power value, the proposed ZFSL-MPAS, SL-MPPAS, and SL-MPGBPPAS methods obtained the lowest BER among all selection approaches in the configurations tested. As can be seen in Figure 2.7a, this benefit is more pronounced when only a few antennas are active. Additionally, ZFSL-MPAS, SL-MPPAS, and SL-MPGBPPAS require less transmission power to achieve a given BER than the other approaches. The proposed channel–level algorithms performed quite similarly to each other. When compared to symbol–level algorithms, channel–level algorithms perform worse since they solely use information about channel estimates. However, symbol-level approaches can be less advantageous as they need to obtain a subset of selected antennas each time a symbol is transmitted. As a result, they must pick antennas more frequently than channel-level methods, increasing the overall computational complexity in this case.

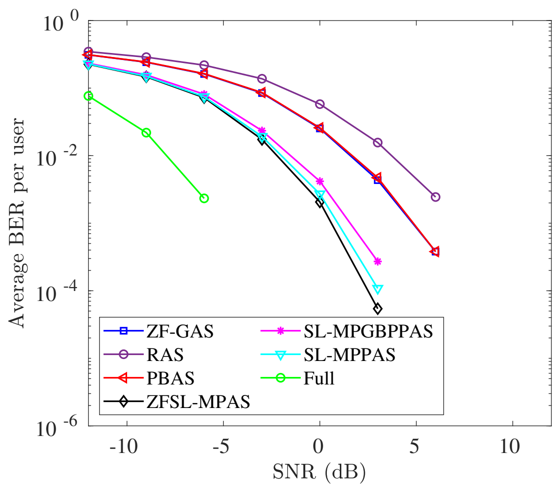
2.8.2 Scenario 2: perfect CSI knowledge, 4-QAM, rich scattering and MR precoding

In scenario 2, the message vector $\mathbf{x}$ is computed using MR precoding to examine the effects of the precoding scheme employed following antenna selection. The results are shown in Figure 2.8.

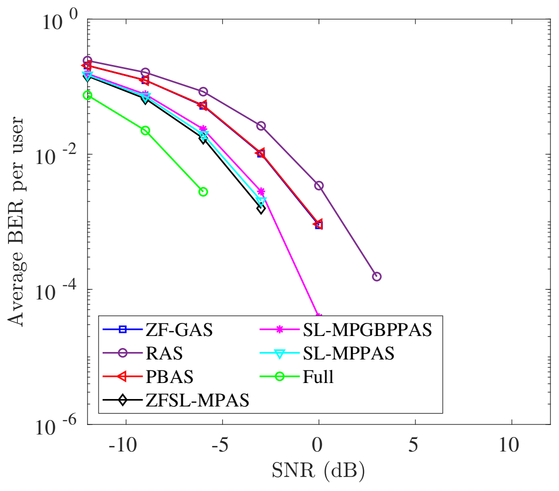
The SL-MPPAS and SL-MPGBPPAS methods that already perform precoding when building the message vector are not reevaluated in this scenario. We conclude that MR precoding is poorer in terms of BER by comparing the findings obtained in Figure 2.7 for ZF precoding and Figure 2.8 for MR precoding. For $S = 12$,



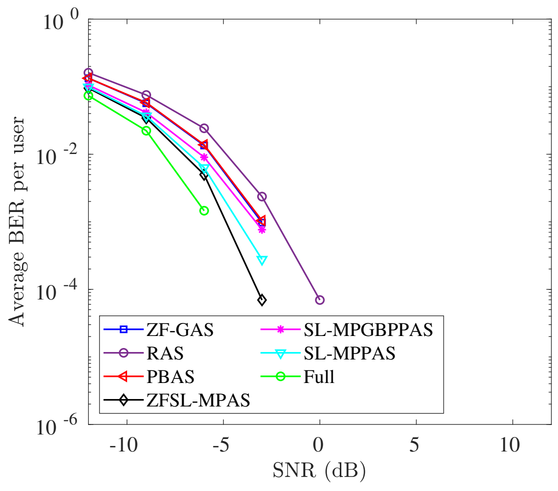
(a) $S = 12$.



(b) $S = 40$.



(c) $S = 100$.



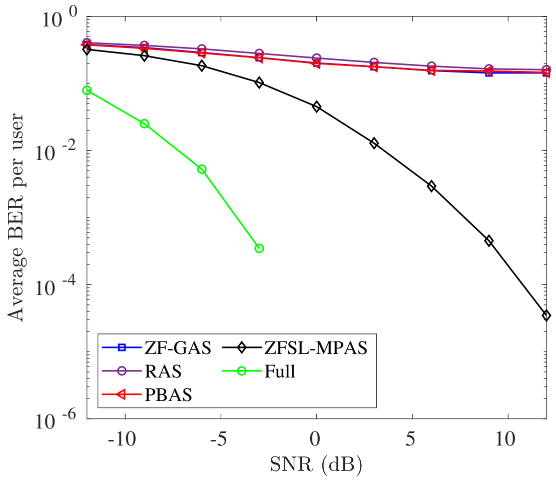
(d) $S = 200$.

Figure 2.7: Scenario 1: average BER per user for a massive MIMO system with perfect CSI knowledge. The channel is modelled considering a rich scattering environment. In this experiment, a base station with $M = 400$ antennas transmits 4-QAM symbols (precoded using ZF precoding scheme) to $K = 12$ terminals using only S antennas active. In this simulation, we consider methods: ZF-GAS (Algorithm 1), ZFSL-MPAS (Algorithm 2), SL-MPPAS (Algorithm 3), SL-MPGBPPAS (Algorithm 4), RAS [1], and PBAS [2].)

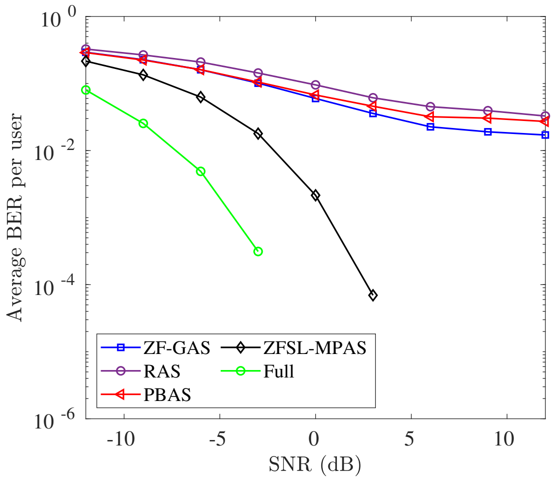
ZFSL-MPAS with MR precoding performs better in terms of BER when compared with the case in which ZF precoding is used. Since $S \ll M$, in this case, matrix $\mathbf{G}_S$ has high sparsity, and hence it is difficult to compute its pseudo-inverse with ZF precoding.

Moreover, depending on the application, MR precoding may be preferable than ZF because it requires less computing effort.

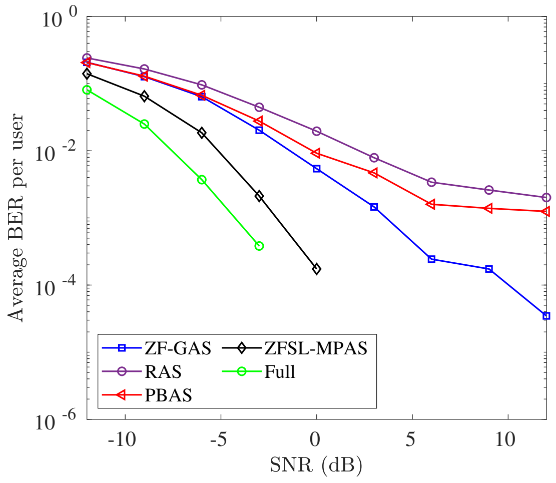
Among the methods evaluated in scenario 2, ZFSL-MPAS achieved the best results.



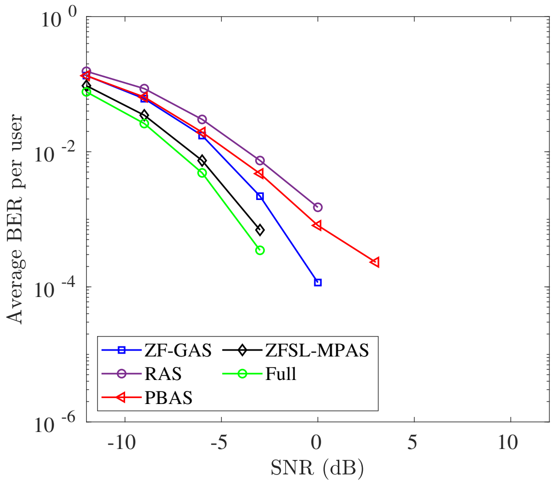
(a) $S = 12$.



(b) $S = 40$.



(c) $S = 100$.

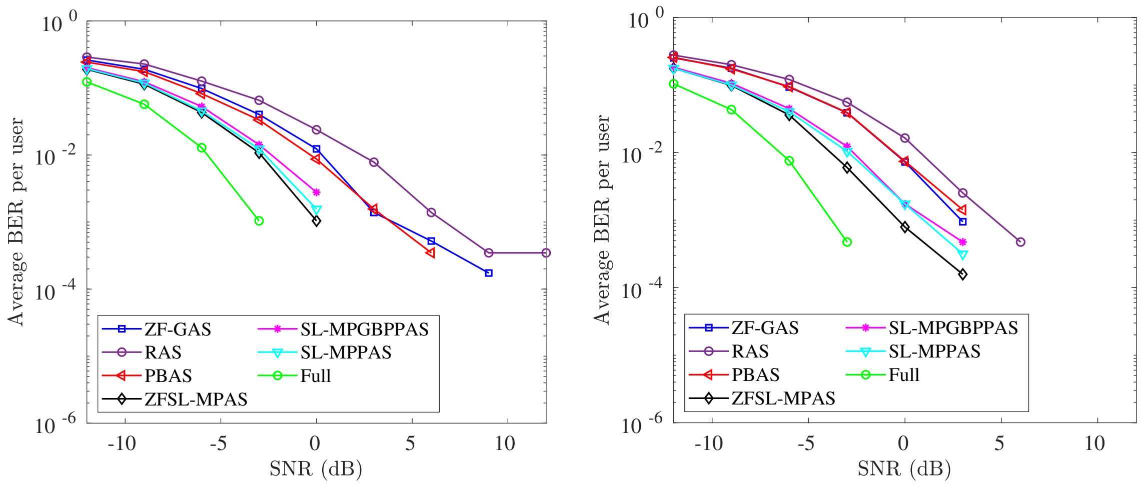


(d) $S = 200$.

Figure 2.8: Scenario 2: average BER per user for a massive MIMO system with perfect CSI knowledge. The channel is modelled considering a rich scattering environment. In this experiment, a base station with $M = 400$ antennas transmits 4-QAM symbols (precoded using MR precoding scheme) to $K = 12$ terminals using only S antennas active. In this simulation, we consider methods: ZF-GAS (Algorithm 1), ZFSL-MPAS (Algorithm 2), RAS [1], and PBAS [2].

2.8.3 Scenario 3: partial CSI knowledge, 4-QAM, and ZF precoding

In scenario 3, we address the case where the BS has partial CSI knowledge, so that the uplink SNR is $\rho_{\text{ul}} = 3$ dB. Figure 2.9 depicts the Average BER per user for scenario 3. As in scenarios 1 and 2, we consider a rich scattering environment in Figure 2.9a. We also consider a poor scattering environment [65] in Figure 2.9b, in which only two paths reach the receivers, which is equivalent to using a correlated Rayleigh fading model with $N = 2$ in equation (2.8). As expected, the overall performance illustrated in Figures 2.9a and 2.9b is less favorable when compared with the case where perfect CSI knowledge is available, as in Figure 2.7c. However, we can observe that even when the channel estimate is not very accurate, the antenna selection algorithms perform reasonably well.

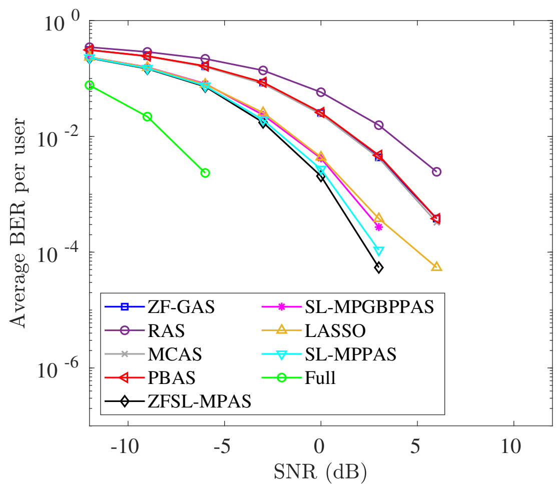


(a) Rich scattering. (b) Poor scattering.

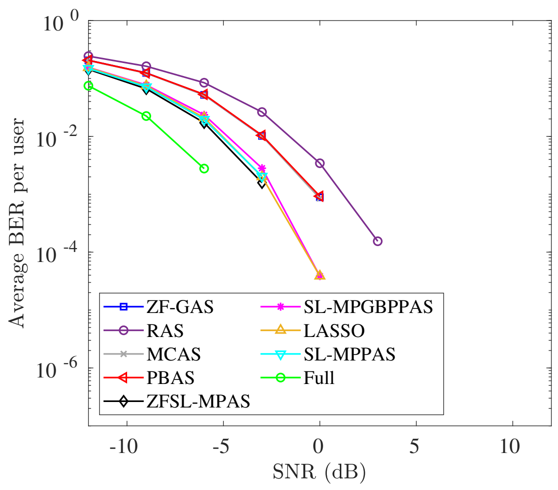
Figure 2.9: Scenario 3: average BER per user for a massive MIMO system with partial CSI knowledge. The channel is modelled considering a rich (a) and poor (b) scattering environment. In this experiment, a base station with $M = 400$ antennas transmits 4-QAM symbols (precoded using ZF precoding scheme) to $K = 12$ terminals using only $S = 100$ antennas active. In this simulation, we consider methods: ZF-GAS (Algorithm 1), ZFSL-MPAS (Algorithm 2), SL-MPPAS (Algorithm 3), SL-MPGBPPAS (Algorithm 4), RAS [1], and PBAS [2].

2.8.4 Scenario 4: perfect CSI knowledge, 4-QAM, rich scattering and ZF precoding: Comparison with other methods

In scenario 4, we compare the proposed antenna selection methods with their counterparts based on convex optimization approaches. We refer to the channel–level method presented in [3] as maximum capacity antenna selection (MCAS). It is worth mentioning that MCAS is also related to the sensor selection approach introduced in [59], as both methods have similar objective functions. Likewise, our proposed symbol–level methods, the least absolute shrinkage, and selection operator (LASSO) proposed in [4] and used in antenna selection in [5] are based on sparse recovery. Nevertheless, LASSO is more related to the SL-MPGBPPAS and SL-MPPAS methods since both generate the precoded signal $\mathbf{x}$ directly. As one can see in Figure 2.10, the proposed antenna selection methods are very close in performance to the algorithms based on convex optimization, with the benefit of being less computationally intense.



(a) $S = 40$.



(b) $S = 100$.

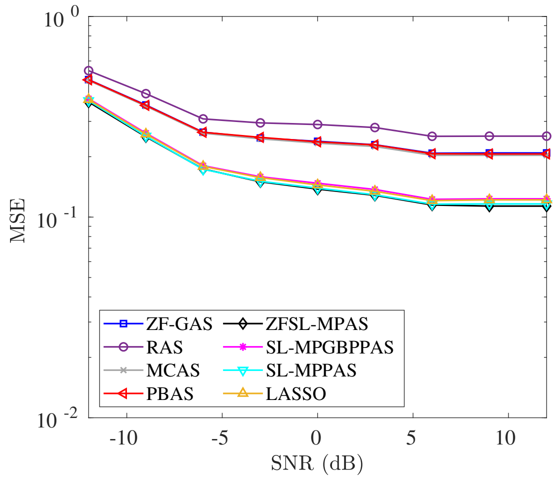
Figure 2.10: Scenario 4: average BER per user for a massive MIMO systems with perfect CSI knowledge. The proposed massive MIMO system is compared with its counterparts in terms of BER. The channel is modelled considering a rich scattering environment. In this experiment, a base station with $M = 400$ antennas transmits 4-QAM symbols (precoded using ZF precoding scheme) to $K = 12$ terminals using only S antennas active. In this simulation we consider methods: ZF-GAS (Algorithm 1), ZFSL-MPAS (Algorithm 2), SL-MPPAS (Algorithm 3), SL-MPGBPPAS (Algorithm 4), RAS [1], PBAS [2], MCAS [3], and LASSO [4, 5].

Since the proposed antenna selection methods are developed aiming at minimizing the MSE between the obtained symbol vector the original symbol vector $\mathbf{q}$, we also present in Figure 2.11 the observed MSE between the obtained symbol vector the original symbol vector plotted against the transmission power for $S \in \{40,100\}$. Moreover, Figure 2.12a depicts the average sum rate, defined in equation (2.38). The downlink SNR is calculated at the terminal using equation (2.39). Figures 2.11 and 2.12a show the advantages of ZFSL-MPAS, SL-MPPAS and SL-MPGBPPAS in terms of MSE and sum rate, respectively.

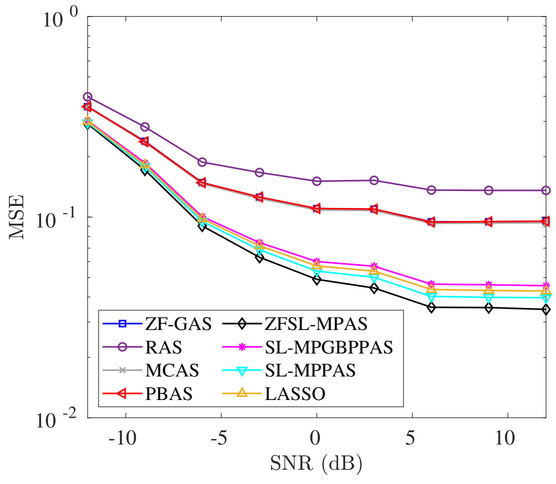
For comparison, Figure 2.12b provides the processing time [66] spent by LASSO, MCAS and the proposed antenna selection algorithm to transmit a block of $L = 25$ messages. A machine with an Intel Core i7-7500U CPU 2.70GHz x4 processor and 7.7 GB of memory was employed. The proposed antenna selection algorithms are less computationally intensive than LASSO, especially the symbol–level ones. Moreover, we verify that the symbol–level algorithms require less computation than a channel-level algorithm such as MCAS.

2.8.5 Scenario 5: perfect CSI knowledge, 16-QAM, rich scattering and ZF precoding.

The average BER per user for scenario 5 is shown in Figure 2.13 for perfect CSI



(a) $S = 40$.



(b) $S = 100$.

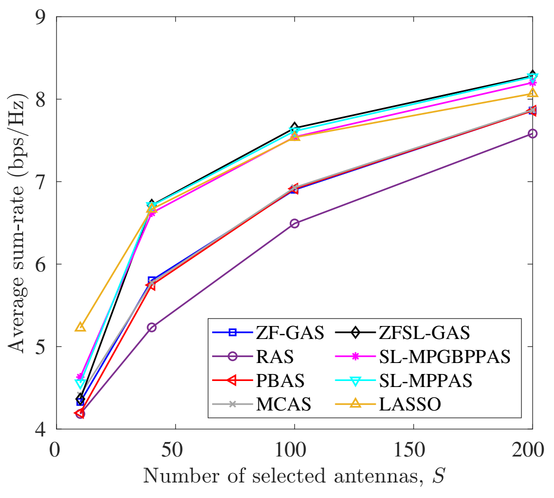
Figure 2.11: Scenario 4: MSE for massive MIMO systems with perfect CSI knowledge. The proposed massive MIMO system is compared with its counterparts in terms of MSE. The channel is modelled considering a rich scattering environment. In this experiment, a base station with $M = 400$ antennas transmits 4-QAM symbols (precoded using ZF precoding scheme) to $K = 12$ terminals using only S antennas active. In this experiment, a base station with $M = 400$ antennas transmits 4-QAM symbols (precoded using ZF precoding scheme) to $K = 12$ terminals using only S antennas active. In this simulation we consider methods: ZF-GAS (Algorithm 1), ZFSL-MPAS (Algorithm 2), SL-MPPAS (Algorithm 3), SL-MPGBPPAS (Algorithm 4), RAS [1], PBAS [2], MCAS [3], and LASSO [4, 5].

condition. As can be verified, the proposed antenna selection methods can also support higher modulations schemes.

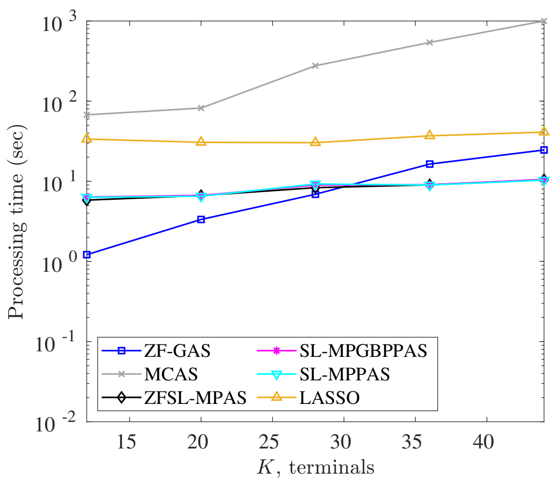
2.9 Conclusions

In this chapter, we proposed an aggregation of antenna selection methods based on matching pursuit and matching pursuit with generalized bit plane techniques. The proposed methods were categorized as a channel–level and symbol–level antenna selection. Although the complexity of the ZF-GAS channel-level method does not justify its performance, its derivation served as a foundation for the symbol-level algorithms. Among the symbol–level methods, ZFSL-MPAS obtains lower BER levels, especially when the SNR is high compared to SL-MPPAS and SL-MPGBPPAS. However, SL-MPPAS and SL-MPGBPPAS do not need to perform external precoding schemes after selection, reducing the computational complexity.

The computational complexity of the proposed antenna selection algorithms was quantified by the number of flops required to compute the transmitted block composed of 25 messages, $\mathbf{x} \in \mathbb{C}^{M\times 1}$. We do this to be fair with the channel–level methods. Moreover, it is worth mentioning that both LASSO and MCAS solutions



(a) Sum rate.

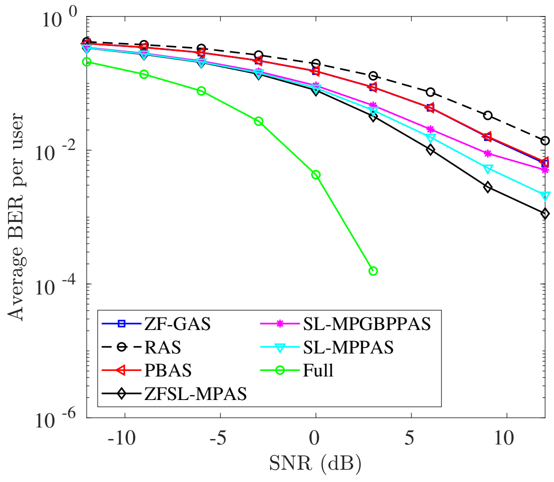


(b) Computational times.

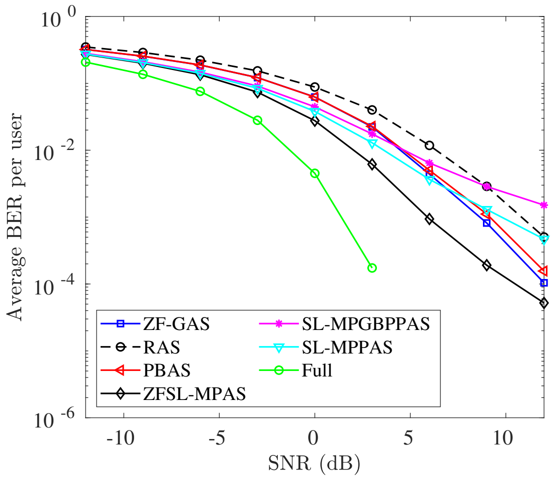
Figure 2.12: Scenario 4: Sum rate and computational times for massive MIMO systems with perfect CSI knowledge. The proposed massive MIMO system is compared with its counterparts in terms of sum rate (a) and computational times (b). The channel is modelled considering a rich scattering environment. In this experiment, a base station with $M = 400$ antennas transmits 4-QAM symbols (precoded using ZF precoding scheme) to $K = 12$ terminals using only S antennas active. In this experiment, a base station with $M = 400$ antennas transmits 4-QAM symbols (precoded using ZF precoding scheme) to $K = 12$ terminals using only S antennas active. In this simulation we consider methods: ZF-GAS (Algorithm 1), ZFSL-MPAS (Algorithm 2), SL-MPPAS (Algorithm 3), SL-MPGBPPAS (Algorithm 4), RAS [1], PBAS [2], MCAS [3], and LASSO [4, 5].

methods are generated with CVX-MATLAB, which can be slow when the number of optimization variables is large.

The proposed algorithms based on the matching pursuit technique were evaluated and compared against the state-of-the-art via bit error rate and MSE as the transmission power was varied. Furthermore, the antenna selection algorithms were tested in different scenarios, comprising rich and poor scattering environments, 4-QAM and 16-QAM modulations, ZF and MR precoders, and perfect and partial CSI knowledge at the BS. The results show that channel–level algorithms are preferred when the coherence time is long, whereas using symbol–level algorithms becomes interesting for short coherence time. The channel–level ZF-GAS algorithm outperforms its counterparts MCAS and PBAS in terms of BER; however, ZF-GAS is more complex than PBAS. In the symbol–level context, the proposed SL-MPPAS and SL-MPGBPPAS algorithms outperform LASSO in terms of BER, with the benefit of requiring fewer computations than LASSO. Furthermore, since the SL-MPGBPPAS algorithm is based on the MPGBP, which obtains a message with quantized elements, it also decreases the demands on amplifier linearity and the BS costs even more.



(a) $S = 40$.



(b) $S = 100$.

Figure 2.13: Scenario 5: average BER per user for a massive MIMO system with perfect CSI knowledge. The channel is modelled considering a rich scattering environment. In this experiment, a base station with $M = 400$ antennas transmits 16-QAM symbols (precoded using ZF precoding scheme) to $K = 12$ terminals using only S antennas active. In this simulation, we consider methods: ZF-GAS (Algorithm 1), ZFSL-MPAS (Algorithm 2), SL-MPPAS (Algorithm 3), SL-MPGBPPAS (Algorithm 4), RAS [1], and PBAS [2].

In the next chapter, we will shift gears and quickly review the machine learning techniques required to develop the solutions for wireless OFDM systems in the remaining chapters. The considered machine learning tools can be grouped as supervised and semi-supervised learning approaches. First, we will consider feedforward neural networks trained in standard and robust modes from supervised machine learning. Robust neural networks are trained using adversarial examples or perturbed versions of clean data samples. From semi-supervised learning, we consider a deep Q neural network to address problems in OFDM in which the training data consists of both labeled and unlabeled observations.

Chapter 3

Machine learning tools

3.1 Introduction

In the previous chapter, we presented greedy algorithms to select a subset of active antennas in a massive MIMO system, aiming at reducing the base station cost. Moreover, the algorithms' greedy nature enabled the optimization problem that defines the antenna selection task to be solved with reduced computational complexity. Before switching from antenna selection solutions for massive MIMO to machine learning (ML) solutions for challenges in OFDM systems, we make a pause to review the ML techniques required to understand better Chapters 5 and 6. Therefore, this chapter briefly discusses the leading machine learning tools considered when developing solutions for wireless OFDM systems. More details can be found in our book chapter [24].

ML is a branch of artificial intelligence (AI) designed to solve complex tasks by learning from available data automatically. ML algorithms are often categorized as supervised, unsupervised, and semi-supervised. Supervised ML models require a dataset with observations or inputs and corresponding labels or outputs. By learning from this labeled dataset, they can predict the output from unseen input data. On the other hand, unsupervised ML models only need the observations. Unsupervised learning aims to find hidden patterns in unlabeled input data that could be used to classify unseen input data. Semi-supervised learning lies between supervised and unsupervised learning, using both labeled and unlabeled training data. Semi-supervised models use unlabeled data to re-prioritize hypotheses obtained from labeled data.

In this work, we consider artificial neural networks (ANNs) operating in supervised learning mode and a semi-supervised learning regime via deep reinforcement learning to solve practical problems in wireless OFDM systems. First, in Section 3.2, we present the main structure of the considered neural networks operating in supervised mode. We also describe how to achieve more robust ML-based solutions with adversarial training. Then, in Section 3.3, we describe the deep reinforcement learning algorithm from semi-supervised learning. Finally, we end the chapter with concluding remarks regarding the different ML techniques discussed in Section 3.4.

3.2 Supervised learning in feedforward artificial neural networks

Supervised learning is based on teaching a model how to obtain the desired output given a training labeled dataset. Among the several algorithms used in supervised learning processes, we can mention artificial neural networks or just neural networks.

Neural networks are interconnected nodes organized in layers that mimic the human brain. The nodes can represent the input, a bias, activation function, or the output, and the connections between nodes are the NN's weights. As input data is fed into the NN, it adjusts its weights by comparing the input with its label using the error backpropagation algorithm. The training process of neural networks is summarized in Subsection 3.2.1. Although neural networks are great at solving challenging tasks in many fields, they are highly vulnerable to adversarial examples. In computer vision, for instance, adversarial examples are perturbed versions of the original samples imperceptible to the human eyes but powerful enough to make the model misclassify the sample with high confidence. In Subsection 3.2.2, we briefly describe adversarial attacks and how to defend against them via adversarial training to obtain reliable models.

3.2.1 Neural networks system Model

The NN is composed of layers $l = 0, 1, 2, \cdots, L$, where $l = 0$ is the input layer, $l = L$ is the output layer, and hidden layers when $0 < l < L$, as illustrated in Figure 3.1. The NN is characterized by its weights or parameters $\boldsymbol{\Theta} = [\boldsymbol{\theta}^{(1)} \cdots \boldsymbol{\theta}^{(L)}]$, where the weight matrix $\boldsymbol{\theta}^{(l)}$ connects layers $l-1$ and l. When training our model, we want to learn a function $f_{\boldsymbol{\Theta}}(\cdot)$ that approximates an unknown process $f(\cdot)$ that maps a data space $\mathcal{X}$ to an output space $\mathcal{Y}$. Due to practical issues, we only have access to a limited dataset composed of input-output pairs

$$\mathcal{D} = \{(\mathbf{x}(1), \mathbf{y}(1)), (\mathbf{x}(2), \mathbf{y}(2)), \cdots, (\mathbf{x}(M), \mathbf{y}(M))\}, \tag{3.1}$$

where $\mathbf{x}(m) \in \mathbb{R}^{N \times 1}$ for $m = 1, \cdots M$.

In regression problems, the output is a single value $\hat{y} = f_{\boldsymbol{\Theta}}(\mathbf{x}) \in \mathbb{R}$ to be compared with the true value $y \in \mathbb{R}$. In classification problems with C classes, the

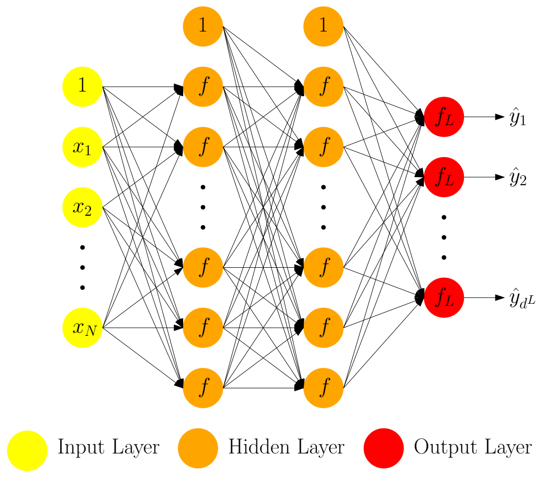


Figure 3.1: Graphical representation of a neural network with 4 layers ($L = 3$).

desired signal $\mathbf{y}(m) \in \mathbb{R}^{C\times 1}$ is one-hot-encoded, meaning that if c is the correct class, $y_c(m) = 1$ and $y_i(m) = 0$ for $i \neq c$. In this case, using the softmax activation function at the output layer is useful since the output signal $\hat{\mathbf{y}} = f_{\mathbf{\Theta}}(\mathbf{x})$ will return a probability distribution on the classes, that is, $\hat{y}_i(m) = P(c = i)$ and the class with greater probability is chosen.

The NN's goal is to find the weights that minimize the loss function $\mathcal{L}(f_{\mathbf{\Theta}}(\mathbf{x}),\mathbf{y})$ as defined in the following optimization problem

$$\min_{\mathbf{\Theta}} \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x},\mathbf{y}\in\mathcal{D}} \mathcal{L}(f_{\mathbf{\Theta}}(\mathbf{x}),\mathbf{y}). \tag{3.2}$$

To do so, we can employ the gradient descent (GD) algorithm with mini-batches. At each iteration $i = 1, 2, \cdots I$ of the mini-batch training, b examples are randomly selected from $\mathcal{D}$ to form batches $\mathcal{B}$ assembled as $\mathbf{X}_{(t,i)} = [\mathbf{x}(1), \mathbf{x}(2), \cdots, \mathbf{x}(b)]$ and $\mathbf{Y}_{(t,i)} = [\mathbf{y}(1), \mathbf{y}(2), \cdots, \mathbf{y}(b)]$ until completing an epoch t. In the forward pass, the input signals in $\mathbf{X}_{(t,i)}$ flow forward through the network and produce the estimates in $\hat{\mathbf{Y}}_{(t,i)}$ to be compared with the truth labels $\mathbf{Y}_{(t,i)}$. In the backward pass, the objective function is minimized with respect to the weights $\mathbf{\Theta}$. By using the GD algorithm, the weights are iteratively updated

$$\mathbf{\Theta}_{k+1} = \mathbf{\Theta}_k - \mu \frac{1}{b} \sum_{\mathbf{x},\mathbf{y}\in\mathcal{B}} \nabla_{\mathbf{\Theta}} \mathcal{L}(f_{\mathbf{\Theta}}(\mathbf{x}),\mathbf{y}), \tag{3.3}$$

where μ is the step size and b is the mini-batch size. We consider the GD algorithm in this work, but lately, more flexible optimizers have been developed, such as Adagrad, Adam, RMSProp, among others [41, 67–70]. Some of these methods use adaptive step sizes that usually achieve better and smoother convergence.

The feedforward process consists of the repetition of two steps in each hidden layer. The first one is the sum of the weighted outputs of the previous layer,

$$\mathbf{a}^{(l)} = \boldsymbol{\theta}^{(l)T}\mathbf{h}^{(l-1)}, \tag{3.4}$$

where $\mathbf{a}^{(l)}$ is the input of layer l. The second step consists of applying an activation function at layer l to obtain the output vector,

$$\mathbf{h}^{(l)} = \begin{bmatrix} 1 \\ f(\mathbf{a}^l) \end{bmatrix}, \text{ for } 1 < l < L-1, \ \mathbf{h}^{(L)} = \begin{bmatrix} f_L(\mathbf{a}^L) \end{bmatrix} \tag{3.5}$$

where $f(\mathbf{a}^{(l)})$ is a vector whose components are $f(a_j^{(l)})$ with $a_j^{(l)} = \sum_{i=0}^{d(l-1)} \theta_{ij}^{(l)} h_i^{(l-1)}$, for $j = 1, \cdots, d^{(l)}$. The input layer is initialized by $\mathbf{h}^{(0)} = [1; \mathbf{x}]^T$ and the feedforward process follows

$$\begin{aligned} \mathbf{x} = \mathbf{h}^0 &\xrightarrow{\boldsymbol{\theta}^{(1)}} \mathbf{a}^{(1)} \xrightarrow{f} \mathbf{h}^{(1)} \xrightarrow{\boldsymbol{\theta}^{(2)}} \mathbf{a}^{(2)} \xrightarrow{f} \mathbf{y}^{(2)} \cdots \\ &\xrightarrow{\boldsymbol{\theta}^{(L)}} \mathbf{a}^{(L)} \xrightarrow{f_L} \mathbf{h}^{(L)} = \hat{\mathbf{y}}. \end{aligned} \tag{3.6}$$

In the backward step, the objective function is minimized with respect to the weights $\boldsymbol{\Theta} = [\boldsymbol{\theta}^{(1)} \cdots \boldsymbol{\theta}^{(L)}]$. By using the gradient descent algorithm, the weights are iteratively updated following the negative gradient direction,

$$\boldsymbol{\theta}^{(l)}(k+1) = \boldsymbol{\theta}^{(l)}(k) - \frac{\mu}{b}\frac{\partial \mathcal{L}(\boldsymbol{\Theta})}{\partial \boldsymbol{\theta}^{(l)}}\Big|_{\boldsymbol{\theta}(k)}, \tag{3.7}$$

where μ is the step size and b is the amount of selected data. Here, the objective function is the sum of the point-wise square error related to each training sample,

$$\mathcal{L}(\boldsymbol{\Theta}) = \frac{1}{M}\sum_{m=1}^{M} \mathcal{L}_m(\boldsymbol{\Theta}) = \frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{d^{(L)}} \mathcal{L}_m^n(\boldsymbol{\Theta}), \tag{3.8}$$

where $\mathcal{L}_m^n$ is the objective function related to the output node n for the sample $\mathbf{x}(m)$. The derivatives in (3.7) are computed recursively in a backward fashion. The chain rule is applied so that the partial derivative is partitioned into two new expressions,

$$\frac{\partial \mathcal{L}_m}{\partial \boldsymbol{\theta}^{(l)}} = \frac{\partial \mathbf{a}^{(l)}}{\partial \boldsymbol{\theta}^{(l)}}\frac{\partial \mathcal{L}_m}{\partial \mathbf{a}^{(l)}} = \mathbf{h}^{(l-1)}\boldsymbol{\delta}^{(l)T} \tag{3.9}$$

where the first term is computed using equation (3.5) and the second term is obtained from the back-propagation process. Vector $\boldsymbol{\delta}^{(l)}$ is known as the sensitivity vector for the layer l, which represents the gradient of the cost $\mathcal{L}_m$ with respect to input $\mathbf{a}^{(l)}$. Once the sensitivity vector $\boldsymbol{\delta}^{(l+1)}$ is obtained, it is multiplied by the weight matrix

$\boldsymbol{\theta}^{(l+1)}$ and the bias component is discarded, resulting in

$$\boldsymbol{\epsilon}^{(l)} = [\boldsymbol{\theta}^{(l+1)}\boldsymbol{\delta}^{(l+1)}]_1^{d^{(l)}}. \tag{3.10}$$

In the following, the sensitivity vector for layer l is

$$\boldsymbol{\delta}^{(l)} = f'(\mathbf{a}^{(l)}) \otimes \boldsymbol{\epsilon}^{(l)}, \tag{3.11}$$

where $f'(\mathbf{a}^{(l)})$ is the derivative of the activation function f in $\mathbf{a}^{(l)}$ and $\otimes$ represents element-wise multiplication. The whole procedure is detailed as

$$\begin{aligned}
\boldsymbol{\delta}^{(L)} &\xrightarrow{\times\boldsymbol{\theta}^{(L)}\big|_1^{d^{(L-1)}}} \boldsymbol{\epsilon}^{(L-1)} \xrightarrow{\otimes f'(\mathbf{a}^{(L-1)})} \boldsymbol{\delta}^{(L-1)} \\
&\xrightarrow{\times\boldsymbol{\theta}^{(L-1)}\big|_1^{d^{(L-2)}}} \cdots \xrightarrow{\times\boldsymbol{\theta}^{(2)}\big|_1^{d^{(1)}}} \boldsymbol{\epsilon}^{(1)} \xrightarrow{\otimes f'(\mathbf{a}^{(1)})} \boldsymbol{\delta}^{(1)},
\end{aligned} \tag{3.12}$$

where "$\big|_1^{d^{(l)}}$" means that only the components $1, 2, \cdots, d^{(l)}$ of the vector $\boldsymbol{\theta}^{(l+1)}\boldsymbol{\delta}^{(l+1)}$ are selected.

3.2.2 Robust neural networks via adversarial training

State-of-the-art DNNs are known to be highly vulnerable to adversarial examples [71, 72]. Adversarial examples are perturbed versions of clean samples that can make, for example, the model misclassifies a panda as an indri with high confidence, as illustrated in Figure 3.2. The added perturbation, however, is imperceptible for us humans.

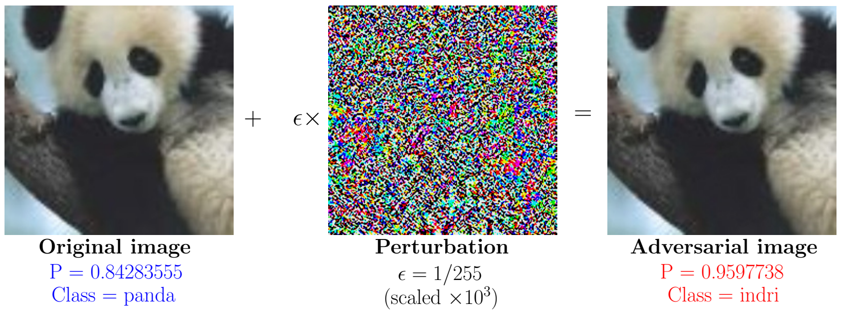


Figure 3.2: Example of adversarial image generation.

Since adversarial attacks might lead to disastrous implications in areas like healthcare [73], climate [74] and finance [75], defending against them is critical. So far, adversarial training is the most effective approach to mitigate the effect of strong attacks like the Projected Gradient Descent (PGD) attack [76], DeepFool [77], and AutoAttack [78]. Training the DNN with perturbed versions of the original samples makes it possible to improve the accuracy on unseen adversarial examples, also known as *robustness accuracy* [79].

Adversarial training continually creates and incorporates adversarial examples into the training process of a neural network classifier

$$f_{\mathbf{\Theta}}(\mathbf{x}) : \mathbb{R}^N \to \{1 \cdots C\}, \tag{3.13}$$

with $\mathbf{\Theta}$ as weights. The NN maps an input image $\mathbf{x}$ to a label $\mathbf{y}$ from a dataset $\mathcal{D}$, defined in equation (3.1), with C possible classes. Adversarial training attempts to solve the min-max optimization problem

$$\begin{aligned} \min\nolimits_{\mathbf{\Theta}} \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x},\mathbf{y} \in \mathcal{D}} \max\nolimits_{\boldsymbol{\eta}} \ \mathcal{L}(f_{\mathbf{\Theta}}(\mathbf{x} + \boldsymbol{\eta}),\mathbf{y}) \\ \text{s.t } ||\boldsymbol{\eta}||_p \leq \epsilon, \end{aligned} \tag{3.14}$$

where $\mathcal{L}(f_{\mathbf{\Theta}}(\mathbf{x}+\boldsymbol{\eta}),\mathbf{y})$ is the loss function on the adversarial sample and $\boldsymbol{\eta}$ is a small perturbation constrained by ϵ.

Creating adversarial samples involves solving the inner maximization problem in equation (3.14), in which the loss function $\mathcal{L}$ is maximized in an effort to change the prediction, that is, $f_{\mathbf{\Theta}}(\mathbf{x}+\boldsymbol{\eta}) \neq f_{\mathbf{\Theta}}(\mathbf{x})$. The optimization constraints ensure that the distance between the adversarial and original example should be less than ϵ under a particular norm, $||\boldsymbol{\eta}||_p \leq \epsilon$. The norms aim to quantify how imperceptible to humans an adversarial example is. Some examples of norms are the l_0 norm, l_2 norm, and l_∞. We then briefly review the most popular methods to create adversarial examples.

Introduced by [72], the Fast Gradient Sign Method (FGSM) attack generates adversarial examples by modifying the input towards the direction where the loss $\mathcal{L}$ increases

$$\begin{aligned} \mathbf{x}' &= \mathbf{x} + \boldsymbol{\eta} \\ &= \mathbf{x} + \epsilon \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{\Theta}\mathbf{x},\mathbf{y})), \end{aligned} \tag{3.15}$$

with sign$(\cdot)$ the sign function, and $\nabla_{\mathbf{x}}\mathcal{L}(\mathbf{\Theta}, \mathbf{x},y)$ the loss gradient with respect to $\mathbf{x}$. One of the strongest l_∞-bounded attacks, the PGD attack [76] tries to solve the inner maximization problem in equation (3.14) following an iterative procedure. At each step i, the adversarial example is updated as

$$\begin{aligned} \mathbf{x}'_i &= \text{clip}_{\mathbf{x}+\epsilon}(\mathbf{x}_{i-1} + \boldsymbol{\eta}) \\ &= \text{clip}_{\mathbf{x}+\epsilon}(\mathbf{x}_{i-1} + \alpha \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{\Theta}, \mathbf{x},\mathbf{y}))), \end{aligned} \tag{3.16}$$

in which function clip$_{\mathbf{x}+\epsilon}(\cdot)$ clips the input at the positions around the predefined perturbation range. In the context of l_2-bounded attacks, Deepfool [77] is an iterative attack optimized for the l_2-norm based on a linear approximation of the classifier. Using geometry concepts, DeepFool searches within the region of the space that describes the classifier's output (polyhedron) for the minimal perturbation that can change the classifier's decision. Among black-box attacks, a one-pixel attack [80] is an l_0-bounded attack that employs differential evolution to create adversarial examples without knowing the network gradients and their parameters. Finally, the AutoAttack [78] method consists of an ensemble of four attacks: two versions of the PGD attack, the targeted version of the Fast Adaptive Boundary (FAB) attack [81] and the black-box Square Attack [82]. Currently, AutoAttack and PGD attacks are the most popular methods to test adversarial robustness. Since the PGD attack is less computationally intense than AutoAttack, we consider the PGD attack in this work. However, other attacks can be used with the proposed data selection.

With the inner maximization problem addressed, the outer minimization problem in equation (3.14) is then solved to find the model parameters that minimize the loss on the generated adversarial examples. The original dataset $\mathcal{D}$ is split into small batches $\mathcal{B}$ and the GD algorithm is employed to update the model parameters

$$\mathbf{\Theta}_{k+1} = \mathbf{\Theta}_k - \mu \frac{1}{|\mathcal{B}|} \sum_{\mathbf{x},y \in \mathcal{B}} \nabla_{\mathbf{\Theta}} \mathcal{L}(f_{\mathbf{\Theta}}(\mathbf{x} + \boldsymbol{\eta}^*), \mathbf{y}), \tag{3.17}$$

where the gradient is evaluated at the maximum point $\boldsymbol{\eta}^*$ found in the inner maximization problem, thanks to Danskin's theorem [83].

3.3 Semi-supervised deep reinforcement learning

Reinforcement learning (RL) is a machine learning algorithm in which an AI agent learns from an environment by interacting with it. The agent acts on the environment. As feedback, it receives the environment's reward and state, as illustrated in Figure 3.3. The agent aims to find the action that maximizes its total reward.

In Q-learning, the Q-value or total reward $Q_t(s,a)$ represents how good is to choose action a for a particular state s at time t. As training progresses, the algorithm stores the current Q-values for all possible combinations of s and a in the Q-table. The best action is then chosen based on the maximum Q-value for a given state. The Q-learning algorithm updates the Q-value $Q_t(s,a)$ as follows

$$Q_{t+1}(s,a) = (1-\alpha)Q_t(s,a) + \alpha(r + \gamma \max_{a'} Q_t(s',a')), \tag{3.18}$$

where s' is the state reached from state s when acting a, r is the immediate reward received when moving from s to s', α is the learning rate, and γ is the discount factor. The maximum Q-value $\max_{a'} Q_t(s',a')$ in the next state, considering all the

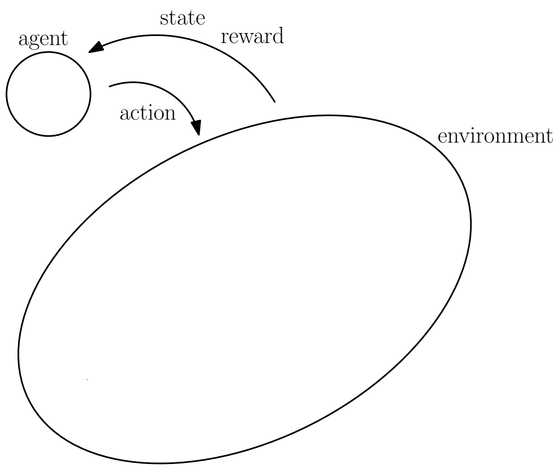


Figure 3.3: Illustration of reinforcement learning.

available actions that lead to the next state, is computed using the Q-table. After some iterations, the Q-table becomes a helpful tool to obtain the best action for a given state.

Nevertheless, the Q-table might become too large, exceeding the computer memory. A deep Q network (DQN) can be used to approximate $Q_t(s,a)$, avoiding the use of a Q-table to remember the solutions. Instead of storing all the Q-values in the Q-table, a neural network is trained to produce different Q-values for each action. The action related to the maximum Q-value is then chosen.

The neural network is trained by minimizing the loss function

$$\mathcal{L} = (r + \gamma \text{max}_{a'} \widehat{Q}(s',a') - Q(s,a))^2, \tag{3.19}$$

where $\text{max}_{a'} \widehat{Q}(s',a')$ is the prediction using the current model on the next state s'.

The RL procedure is illustrated in Figure 3.4, in which the ϵ-greedy policy dictates how to learn the problem. In the beginning, the exploration rate ϵ employed by the ϵ-greedy policy is high, so we uniformly select possible actions. The actions and associated states are stored in the memory replay, which is then used to train the neural network. As the training progresses, the exploration rate ϵ decays, and the action is chosen using the neural network more often.

3.4 Conclusions

In this chapter, we summarized the main machine learning techniques used for developing the OFDM solutions proposed in this work. From supervised learning, we consider both standard and robust feedforward neural networks. Robust here stands for neural networks trained with adversarial examples. Adversarial training is the most popular solution to mitigate the effect of malicious attacks on neural networks.

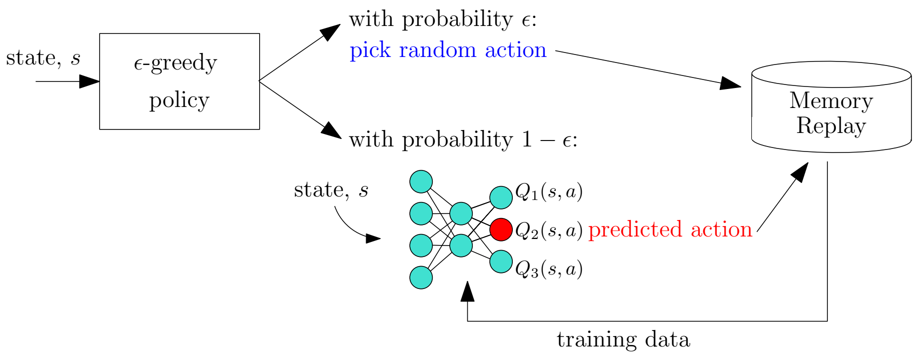


Figure 3.4: Illustration of ϵ-greedy policy.

Although adversarial training can improve the robustness accuracy, it usually sacrifices standard accuracy. The following chapters will discuss a tradeoff regarding standard or adversarial training. From a semi-supervised learning framework, we consider a deep reinforcement learning algorithm to deal with the case in which the training data has both labeled and unlabeled observations. As it will be seen in the following chapters, it is an excellent technique when we need to deliver an adaptive solution for a time-varying system.

By introducing data selection techniques to enhance the performance in regression and classification issues, we narrow down the field of neural networks, which is discussed in this chapter in the following chapter. Furthermore, the proposed data selection strategy helps improving accuracy performance when the neural network is trained in both traditional and robust approaches, as illustrated in the next chapter.

Chapter 4

Data selection and machine learning

4.1 Introduction

The feedforward neural networks (NN) were detailed in the previous chapter to familiarize the reader with the main methods used to create the solutions for wireless OFDM systems in Chapters 5 and 6. In this chapter, we dig into NN training and present data selection strategies to improve their performance. We apply the data selection technique in standard and robust trained neural networks considering different tasks such as temperature prediction and image classification.

The data selection approach was first conceived for linear adaptive filtering and kernel adaptive filtering areas [84–87]. The approach considers the data's relevance during the learning process at each iteration of the parameter update. Inspired by this data-dependency exploitation in adaptive filters, we proposed to select the most informative data samples to compose the batches used for training neural networks [18, 19].

First, we detail the standard training with data selection strategy in Section 4.2, and it mainly comprises the works in [18, 19]. As claimed in [20], such a proper batch composition is also important in the adversarial training domain to alleviate the robustness-accuracy tradeoff. The data selection strategy is then discussed for adversarial training in Section 4.3, and it consists of the contributions in [20]. Finally, we include some concluding remarks regarding the benefits of data selection strategy in Section 4.4.

4.2 Data selection in neural networks

The quantity of information generated worldwide is soaring, raising the question of whether all data stored is useful. Furthermore, the current trend of creating data at an increasing rate is overwhelming the capacity to store, analyze, and make proper use of learning algorithms. In part, this phenomenon originates from the proliferation of sensors, human-computer interactions, internet of things, medical data, and machine-to-machine and mobile communications are a few data generators. Training the Neural Network (NN) algorithms [88–90] requires a large amount of data, leading to high computational costs and unprecedented storage demands for learning machines. Therefore, enormous success in performance is obtained at the expense of increased computational complexity and storage demand, as the amount of data has been growing exponentially. Although modern GPUs have accelerated computation, deep learning is still challenging for limited resource devices and real-time applications.

On the processing side, it is crucial to employ new strategies to alleviate the training burden and increase the success of a learning machine. For example, one way to reduce network complexity is to explore parameter redundancy in the neural network by pruning a portion of the network units. The neurons to be pruned can be randomly selected as in the well-known dropout method [91], or more sophisticated criteria can be employed [92–94].

As DNNs retain a considerable amount of data, another way to manage the excess of network resources is to study data redundancy, i.e., search for data that do not bring relevant information to the training process. For example, data selection is explored in semi-supervised strategies to obtain a subset of unlabeled data based on estimated confidence measures derived from the few available labeled data [95]. This procedure is essential due to the increasing amount of irrelevant information available for processing.

In this sense, we can access the quality of the data being acquired or available to avoid the hurdle brought about by data considered less informative and outliers by all means. There are several challenges to consider in the data selection: What is the performance degradation if bad data is utilized for training a highly complex learning machine? In this case, outlying the data samples considered non-informative (or bad) is a crucial asset to generating a successful machine learning solution. Then, we should go one step further and ask ourselves: Are all data relevant? In most cases, they are not!!! So what is the sub-set of data samples that are just less informative? By extending the data selection method designed for adaptive filtering and kernel adaptive filtering [84–86], we designed a data selection strategy applied at each mini-batch iteration in [19]. The method considers the relevance of the data utilized in the learning process at each iteration related to the parameter update.

In this section, a data selection method in NN is proposed for regression and classification problems [96–98]. The considered decision criterion to update the coefficients is directly related to the objective function. In the neural network framework, each output neuron produces an estimate compared to the target signal and hence generates an error measure defined according to the objective function. As a rule, the closer to zero the error $e(k)$ is, the less informative or relevant will be the contribution of the pair $(\mathbf{x}(k), \mathbf{y}(k))$ to the parameter update at iteration k. Considering all the output neurons, the error vector for the data pair $(\mathbf{x}, \mathbf{y})$ is

$$\mathbf{e}(\hat{\mathbf{y}}, \mathbf{y}) = [e(\hat{y}_1,y_1), e(\hat{y}_2,y_2), \cdots, e(\hat{y}_{d^{(L)}},y_{d^{(L)}})], \tag{4.1}$$

where $\mathbf{y} = [y_1, y_2, \cdots, y_{d^{(L)}}]$ is the target signal and $\hat{\mathbf{y}} = [\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_{d^{(L)}}]$ is the signal estimated by the neural network. The total sum of this error is then expressed as

$$E(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{n=1}^{d^{(L)}} e(\hat{y}_n,y_n). \tag{4.2}$$

Figure 4.1 illustrates the data selection strategy applied in regression and classification of NN problems. At each iteration per epoch, a mini-batch set composed of b data samples illustrated in yellow is used in the forward propagation of the data information. We apply the data selection using equation (4.2) to eliminate the non-informative data represented by the white color. We then proceed with the remainder data in blue in the backpropagation to update the weight vector.

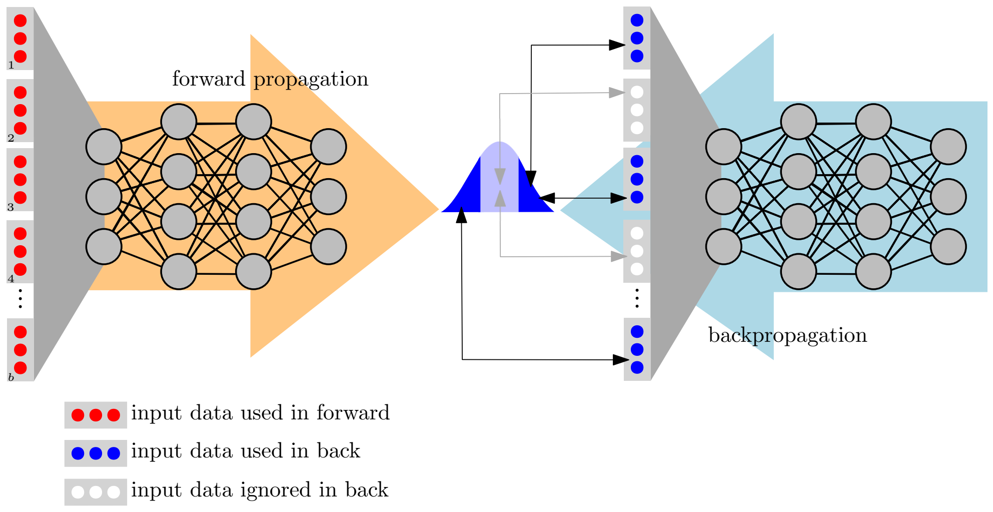

1
2
3
4
⋮
b
forward propagation
⋮
backpropagation
●●● input data used in forward
●●● input data used in back
○○○ input data ignored in back

Figure 4.1: Data selection neural network diagram.

We formulate the data selection method for regression and classification problems in the following. The common choice of the output activation function and the objective function is essential in both. In addition to simplifying the computations in updating weights, we can also improve algorithm performance. In regression problems, the output activation function is commonly linear, and the objective function is the mean squared error. In contrast, for multi-class classification problems, softmax

$$\text{Softmax}(\mathbf{x}) = \frac{\exp(\mathbf{x})}{\sum_{j=1}^{N} \exp(x_j)}, \text{ for } \mathbf{x} \in \mathbb{R}^{N\times 1}, \tag{4.3}$$

and ReLU

$$\text{RelU}(x) = max\{0,x\} \tag{4.4}$$

are used as output activation functions, and cross entropy is used as an objective function.

4.2.1 Regression

The chosen objective function is the mean squared error (MSE),

$$\mathcal{L}(\boldsymbol{\Theta}, \hat{y},y) = \frac{1}{M}\sum_{m=1}^{M}\left[\frac{1}{2}(\hat{y}(m) - y(m))^2\right], \tag{4.5}$$

where the output activation function is linear. Since the output has only one dimension in regression problems, equation (4.2) can be rewritten as

$$E(\hat{y}, y) = e(\hat{y}, y) = y - \hat{y}, \tag{4.6}$$

where y is the desired value, and $\hat{y}$ is the estimated value.

At each epoch t, we assume that the error signal has Gaussian distribution,

$$e \sim \mathcal{N}(0, (\sigma_e^t)^2), \tag{4.7}$$

where $(\sigma_e^t)^2$ is the error variance. By normalizing this error distribution, we obtain

$$\frac{e}{\sigma_e^t} \sim \mathcal{N}(0,\ 1). \tag{4.8}$$

The error variance is calculated using all the training samples in the batch corresponding to the iteration. As in the adaptive filter data selection case [84, 85], the decision criterion to update the coefficients $\boldsymbol{\Theta}$ at iteration i of epoch t can be incorporated into the objective function (4.5)

$$\mathcal{L}_i^1(\boldsymbol{\Theta}, \hat{y},y) = \begin{cases} \frac{1}{2}(e(\hat{y}, y))^2, \text{ if } \sqrt{\tau} \leq \frac{|e(\hat{y},y)|}{\sigma_e^t} \\ 0, \text{ otherwise}, \end{cases} \tag{4.9}$$

where $\sqrt{\tau}$ is the error threshold. Therefore, the objective function is only relevant if the normalized error is above this threshold. Relevant here means that the data sample results in an error signal greater enough to contribute to the learning process.

The data selection method proposes to detect the non-informative values at each iteration after the feedforward propagation process, eliminating the irrelevant data before the backpropagation process, thus reducing the computational cost in the NN algorithm. The prescribed portion of samples considered in the backpropagation process is defined as P_{up}, and it can be computed as

$$P_{\text{up}} = P\left\{\frac{|e|}{\sigma_e^t} > \sqrt{\tau}\right\} = 2Q_e(\sqrt{\tau}), \tag{4.10}$$

where $Q_e(\cdot)$ is the complementary Gaussian cumulative distribution function [99], given by

$$Q_e(x) = \frac{1}{\sqrt{2\pi}}\int_x^{\infty} \exp(-t^2/2)dt. \tag{4.11}$$

In this way, the threshold $\sqrt{\tau}$ associated with a prescribed P_{up} can be obtained from equation (4.10) as

$$\sqrt{\tau} = Q_e^{-1}\left(\frac{P_{\text{up}}}{2}\right), \tag{4.12}$$

where $Q_e^{-1}(\cdot)$ is the inverse of the $Q_e(\cdot)$ function.

During the mini-batch of b samples, we obtain the selected set

$$\mathcal{R} \rightarrow \begin{cases} k \notin \mathcal{R}, \text{ if } \frac{|e(\hat{y}(k),\bar{y}(k))|}{(\sigma_e^t(i))} \leq \sqrt{\tau} \\ k \in \mathcal{R}, \text{ otherwise} \end{cases}, \tag{4.13}$$

for $k = 1, \cdots, b$ and it is composed of the indexes of the selected data $\mathcal{R} = [k_1, k_2 \cdots, k_p]$.

Since we are selecting an estimated portion $\hat{P}_{\text{up}}$ from the training dataset with a mini-batch of b samples, the weight update equation is modified to

$$\boldsymbol{\theta}^{(l)}(k+1) = \boldsymbol{\theta}^{(l)}(k) - \frac{\mu}{b\hat{P}_{\text{up}}}\mathbf{h}_{\mathcal{R}}^{(l-1)}\boldsymbol{\delta}_{\mathcal{R}}^{(l)^T}, \tag{4.14}$$

where $\mathcal{R}$ is the selected set to update the weights at iteration i, $\hat{P}_{\text{up}} = \frac{|\mathcal{R}|}{b}$, $\mathbf{h}_{\mathcal{R}}^{(l-1)}$ is the output of layer $l-1$, and $\boldsymbol{\delta}_{\mathcal{R}}^{(l)^T}$ is the sensitivity vector of layer l.

At iteration i of epoch t, the estimated error variance is calculated by

$$(\sigma_e^t(i))^2 = (1-\lambda_e)\sigma_e^2 + (\lambda_e)(\sigma_e^t(i-1))^2, \tag{4.15}$$

where σ_e^2 is the error variance related to the data in the i-th iteration, and λ_e is a forgetting factor. At the start of epoch t, the estimated error variance depends on the last error variance $(\sigma_e^{t-1}(b))^2$ from the previous epoch, thus the equation for this dependence is established by

$$(\sigma_e^t(0))^2 = P_{\text{up}}(\sigma_e^{t-1}(b))^2. \tag{4.16}$$

The main goal of the data selection in NN is to reduce the computational cost, with the possibility of improving algorithm performance. The data selection NN algorithm for a regression problem is outlined in Algorithm 5. For each iteration i in the algorithm, it is considered only a subset of data of size b known as mini-batch to adapt the NN. As shown in Algorithm 5, at each epoch, one can compute the objective function for the training dataset J_{train} and also for the validation dataset J_{val}, if this set is previously defined.

4.2.2 Classification

For classification problems, we consider the cross entropy as the objective function

$$\mathcal{L}(\mathbf{\Theta}, \hat{\mathbf{y}}, \mathbf{y}) = -\frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{d^{(L)}} y_n(m)\log(\hat{y}_n(m)), \tag{4.17}$$

where $d^{(L)}$ is the number of classes. By considering a classification problem with only two classes, equation (4.17) becomes

$$\mathcal{L}(\mathbf{\Theta}, \hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{M}\sum_{m=1}^{M}\left[-y_0(m)\log(\hat{y}_0(m)) - (1-y_1(m))\log(1-\hat{y}_1(m))\right]. \tag{4.18}$$

For multi-class problems, the labels are one-hot encoded. Therefore, by excluding the summing elements that are zero due to target labels, and considering the softmax activation function in equation (4.3), equation (4.17) becomes

$$\mathcal{L}(\mathbf{\Theta}, \hat{\mathbf{y}}, \mathbf{y}) = -\frac{1}{M}\sum_{m=1}^{M}\log\left(\frac{\exp(y_c(m))}{\sum_{j=1}^{d^{(L)}}\exp(\hat{y}_c(m))}\right), \tag{4.19}$$

where c is the index of the target class. Since $\log(u/v) = \log(u) - \log(v)$, equation (4.19) can be rewritten as

$$\mathcal{L}(\mathbf{\Theta}, \hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{M}\sum_{m=1}^{M}\log\left(\sum_{c=1}^{d}{}^{(L)}\exp(\hat{y}_c(m))\right) - y_c(m), \tag{4.20}$$

where $d^{(L)}$ is the number of classes.

The error $e(\hat{y}_n, y_n)$ in equation (4.1) is defined for the mean squared error and

Algorithm 5 Data selection (DS) feedForward multilayer neural network (NN) algorithm in a regression problem

```
1: Initialize dataset: ${(\mathbf{x}(1), \mathbf{y}(1)), (\mathbf{x}(2), \mathbf{y}(2)), \cdots, (\mathbf{x}(M), \mathbf{y}(M))}$, and weights: $\mathbf{\Theta} = \{\boldsymbol{\theta}^{(1)},\boldsymbol{\theta}^{(2)}, \cdots ,\boldsymbol{\theta}^{(L)}\}$ (random matrices)
2: Select step size $\mu > 0$, number of epochs $n_{\text{ep}}$, mini-batch size $b$, number of layers $L$, number of nodes $(d^{(1)}, \cdots, d^{(L)})$, activation function $f$, output function $f_L$, and forgetting factor $\lambda_e$
3: Define objective function: $\mathcal{L}$ = Mean squared error, number of iterations: $I = M/b$, and initial error variance: $\sigma_e^0(b) = 0$
4: Prescribe desired probability of update: $P_{\text{up}}$ and threshold: $\sqrt{\tau} = Q_e^{-1}\left(\frac{P_{\text{up}}}{2}\right)$
5: for $t = 1 : n_{\text{ep}}$ (for each epoch) do
6:     $(\sigma_e^t(0))^2 = P_{\text{up}}(\sigma_e^{t-1}(b))^2$
7:     for $i = 1 : I$ (for each iteration) do
8:         Randomly select $b$ samples in the training dataset
9:         $\mathbf{X}_{(t,i)} = [\bar{\mathbf{x}}(1), \bar{\mathbf{x}}(2), \cdots, \bar{\mathbf{x}}(b)]$, $\mathbf{Y}_{(t,i)} = [\bar{\mathbf{y}}(1), \bar{\mathbf{y}}(2), \cdots, \bar{\mathbf{y}}(b)]$
10:        Forward propagation:
11:        $\mathbf{H}^{(0)} = [h^{(0)}(1),h^{(0)}(2), \cdots, h^{(0)}(b)] = [ones(1, b);$
12:        $\mathbf{X}_{(t,i)}]$ ($ones(1, b)$ is the bias term)
13:        for $l = 1 : L - 1$ do
14:            $\mathbf{A}^{(l)} = \boldsymbol{\theta}^{(l)^T}\mathbf{H}^{(l-1)}$
15:            $\mathbf{H}^{(l)} = [ones(1,b); f(\mathbf{A}^{(l)})]$
16:        end for
17:        $\mathbf{A}^{(L)} = (\boldsymbol{\theta}^{(L)})^T\mathbf{H}^{(L-1)}$
18:        $\hat{\mathbf{Y}}_{(t,i)} = [\hat{y}(1), \hat{y}(2), \cdots, \hat{y}(b)] = \mathbf{H}^{(L)} = f_L(\mathbf{A}^{(L)})$
19:        Data selection:
20:        $e(\hat{y}_n(k), \bar{y}_n(k)) = \begin{cases} (\hat{y}_n(k) - \bar{y}_n(k))^2, \text{ if MSE} \\ -\bar{y}_n(k)\ln(\hat{y}_n(k)) - (1 - \bar{y}_n(k))\ln(1 - \hat{y}_n(k)), \text{ if CE} \end{cases}$
21:         for $k = 1, \cdots, b$ and $n = 1, \cdots, d^{(L)}$
22:        $\mathcal{E}^n = [e(\hat{y}_n(1), \bar{y}_n(1)), \cdots, e(\hat{y}_n(b), \bar{y}_n(b))]$,        for $n = 1, \cdots, d^{(L)}$
23:        $\boldsymbol{E} = \left[\sum_{n=1}^{d^{(L)}} e(\hat{y}_n(1), \bar{y}_n(1)), \cdots, \sum_{n=1}^{d^{(L)}} e(\hat{y}_n(b), \bar{y}_n(b))\right]$
24:        $t_{\text{bin}} \sim \text{Bin}(n,p)$,
25:        $\mathcal{C} = [k_1, k_2 \cdots, k_{t_{\text{bin}}}]$
26:        $\hat{P}_{\text{up}} = |\mathcal{C}|/b$ where $\mathcal{C}$ is the index set related to the $t_{\text{bin}}$ largest values in
   vector $\boldsymbol{E}$
27:        $\mathbf{Y}_{\mathcal{C}} = [\bar{\mathbf{y}}(k_1), \cdots, \bar{\mathbf{y}}(t_{\text{bin}})]$, $\hat{\mathbf{Y}}_{\mathcal{C}} = [\hat{\mathbf{y}}(k_1), \cdots, \hat{\mathbf{y}}(t_{\text{bin}})]$
28:        Backpropagation:
29:        $\mathbf{\Delta}_{\mathcal{R}}^{(L)} = [\boldsymbol{\delta}^{(L)}(k_1),\boldsymbol{\delta}^{(L)}(k_2), \cdots, \boldsymbol{\delta}^{(L)}(k_p)] = f_L'(\mathbf{A}_{\mathcal{R}}^{(L)}) \otimes (\hat{\mathbf{Y}}_{\mathcal{R}} - \mathbf{Y}_{\mathcal{R}})$
30:        for $l = L - 1 : -1 : 1$ do
31:            $\mathbf{\Delta}_{\mathcal{C}}^{(l)} = f'(\mathbf{A}_{\mathcal{C}}^{(l)}) \otimes [\mathbf{W}^{(l+1)}\mathbf{\Delta}_{\mathcal{C}}^{(l+1)}]_1^{d^{(l)}}$
32:        end for
33:        Updating the weights:
34:        for $l = 1 : L$ do
35:            $\boldsymbol{\theta}^{(l)} = \boldsymbol{\theta}^{(l)} - \frac{\mu}{b\hat{P}_{\text{up}}}\mathbf{H}_{\mathcal{C}}^{(l-1)}(\mathbf{\Delta}_{\mathcal{C}}^{(l)})^T$
36:        end for
37:    end for
38:    $\mathcal{L}_{\text{train}}(\mathbf{\Theta}, \hat{\mathbf{y}},\mathbf{y}) = \frac{1}{M}\sum_{m=1}^{M}\sum_{n=1}^{d^{(L)}}\mathcal{L}_m^n(\mathbf{\Theta}, \hat{y},y)$(and  $\mathcal{L}_{\text{val}}$ if previously defined)
39: end for
```

binary cross entropy, respectively, as

$$\begin{aligned} e_{\text{MSE}}(\hat{y}_n, y_n) &= (\hat{y}_n - y_n)^2, \text{and} \\ e_{\text{CE}}(\hat{y}_n, y_n) &= -y_n \ln(\hat{y}_n) - (1 - y_n) \ln(1 - \hat{y}_n), \end{aligned} \tag{4.21}$$

where $\hat{y}_n$ is the estimated output for the n-th class and y_n is the n-th desired value for the output $\mathbf{y}$.

In classification problems, the last layer has multiple outputs corresponding to multiple classes, so it is difficult to infer a distribution for the error signal, as in the regression problem. Hence, the error distribution is not normalized by its variance, and we directly use equation (4.2).

Also, due to the difficulty in obtaining this probabilistic function, we propose another procedure with a similar idea to obtain the threshold. In classification problems, the data selection aims to detect the error values in equation (4.2) smaller than a given threshold, such that the related errors are disregarded in the process of updating the coefficients. This proposal requires selecting a threshold for each mini-batch per iteration, chosen from a binomial distribution with $n = b$ and $p = P_{\text{up}}$,

$$t_{\text{bin}} \sim \text{Bin}(n,p). \tag{4.22}$$

The data associated with measures smaller than the t_{bin}-th largest components of $E(\hat{\mathbf{y}},\mathbf{y})$ in equation (4.2) are eliminated before the backpropagation process. Defining $\mathcal{C}$ as the set of indexes k of the components of $E(\hat{\mathbf{y}},\mathbf{y})$ greater or equal to the t_{bin}-th largest value of $E(\hat{\mathbf{y}},\mathbf{y})$, we obtain

$$\mathcal{C} = [k_1, k_2, \cdots, k_{t_{\text{bin}}}]. \tag{4.23}$$

These t_{bin} data samples are the subset considered in backpropagation. We also define the set of size $b - t_{\text{bin}}$ containing the indexes of data examples that lead to the smallest error signals in $E(\hat{\mathbf{y}},\mathbf{y})$,

$$\mathcal{B} = [k_1, k_2, \cdots, k_{b-t_{\text{bin}}}]. \tag{4.24}$$

The data examples whose indexes are in set $\mathcal{B}$ are eliminated before the backpropagation process.

We also consider a fine adjustment factor $0 < \alpha \leq 1$ to randomly select $\alpha \times 100\%$ of the examples in set $\mathcal{B}$ to be definitely eliminated. The remaining $(1 - \alpha) \times 100\%$ of the examples in set $\mathcal{B}$ are temporarily included in set $\mathcal{C}$. Then, we randomly select t_{bin} samples to keep in set $\mathcal{C}$. This fine adjustment factor α can be useful when the dataset is too complex, so eliminating the smallest errors can impair the final results.

As in the regression problem, this data selection method aims at identifying the non-informative values after the feedforward propagation process at each iteration, eliminating some of the data before the backpropagation process. As we are selecting an estimated portion $\hat{P}_{\text{up}}$ of data in each iteration, the update equation is rewritten as

$$\boldsymbol{\theta}^{(l)}(k+1) = \boldsymbol{\theta}^{(l)}(k) - \frac{\mu}{b\hat{P}_{\text{up}}}\mathbf{h}_{\mathcal{C}}^{(l-1)}\boldsymbol{\delta}_{\mathcal{C}}^{(l)^T}, \tag{4.25}$$

where $\hat{P}_{\text{up}} = \frac{|\mathcal{C}|}{b}$.

The complete procedure for Data Selection FeedForward Multilayer Neural Network is described in Algorithm 6. For each iteration i in Algorithm 6, it is considered only a subset of data of size b known as mini-batch to adapt the NN. As in the regression problem, at each epoch, one can compute the objective function for the training dataset $\mathcal{L}_{\text{train}}$, and also for the validation dataset $\mathcal{L}_{\text{val}}$, if this set is previously defined.

4.2.3 Computational complexity

In this subsection, we quantify the complexity of the data selection method by counting the required number of flops in the forward and backward steps of a NN with four layers (two hidden layers). Consider i as the number of nodes of the input layer, j the number of nodes in the second layer, k the number of nodes in the third layer, and l the number of nodes in the output layer. The parameters are denoted as n_{ep} (epoch), b (mini-batch size), I (number of iterations), P_{up} (probability of update).

As previously defined in Section 2.7 of Chapter 2, a flop is a floating-point operation [100]. Real addition, subtraction, multiplication, division, and exponential are counted as one flop.

At each iteration for a particular epoch, we have b training samples. In the forward process, the following operations are performed from the current layer to the next layer

$$\begin{aligned} \mathbf{A}^{(l)} &= (\mathbf{W}^{(l)})^T\mathbf{H}^{(l-1)}, \\ \mathbf{H}^{(l)} &= [ones(1,b); f(\mathbf{A}^{(l)})]. \end{aligned} \tag{4.26}$$

From the first layer to the second layer, the first operation is the product between two matrices $\mathbf{W}^{(1)} \in \mathbb{R}^{i\times j}$ and $\mathbf{H}^{(0)} \in \mathbb{R}^{i\times b}$, which consist of jb inner products between vectors of size i. This inner product involves $i-1$ additions and i multiplications. Then, the resulting number of flops is $(2i-1)jb$. The activation function (ReLU) applied in the second equation has 0 flops. Then, we have $(2i-1)jb$ flops.

Algorithm 6 Data selection (DS) feedForward multilayer neural network algorithm in a classification problem

```
1: Initialize dataset: ${(\mathbf{x}(1),\mathbf{y}(1)),(\mathbf{x}(2),\mathbf{y}(2)),\cdots,(\mathbf{x}(M),\mathbf{y}(M))}$, and weights: $\boldsymbol{\Theta} = \{\boldsymbol{\theta}^{(1)},\boldsymbol{\theta}^{(2)},\cdots,\boldsymbol{\theta}^{(L)}\}$ (random matrices)
2: Select step size $\mu > 0$, number of epochs $n_{\text{ep}}$, mini-batch size $b$, number of layers $L$, number of nodes $(d^{(1)},\cdots,d^{(L)})$, activation function $f$, output function $f_L$, and forgetting factor $\lambda_e$
3: Define objective function:
4: Option 1: Objective function $\mathcal{L}$ = Mean squared error (MSE), and output function $f_L$ = Linear or Hyperbolic tangent;
5: Option 2: Objective function $\mathcal{L}$ = cross entropy (CE), and output function $f_L$ = Softmax;
6: Prescribe desired probability of update: $P_{\text{up}}$
7: for $t = 1 : n_{\text{ep}}$ (for each epoch) do
8:     $(\sigma_e^t(0))^2 = P_{\text{up}}(\sigma_e^{t-1}(b))^2$
9:     for $i = 1 : I$ (for each iteration) do
10:        Randomly select $b$ samples in the training dataset
11:        $\mathbf{X}_{(t,i)} = [\bar{\mathbf{x}}(1),\bar{\mathbf{x}}(2),\cdots,\bar{\mathbf{x}}(b)]$, $\mathbf{Y}_{(t,i)} = [\bar{\mathbf{y}}(1),\bar{\mathbf{y}}(2),\cdots,\bar{\mathbf{y}}(b)]$
12:        Forward propagation:
13:        $\mathbf{H}^{(0)} = [h^{(0)}(1),h^{(0)}(2),\cdots,h^{(0)}(b)] = [ones(1,b);$
14:        $\mathbf{X}_{(t,i)}]$ ($ones(1,b)$ is the bias term)
15:        for $l = 1 : L-1$ do
16:            $\mathbf{A}^{(l)} = \boldsymbol{\theta}^{(l)^T}\mathbf{H}^{(l-1)}$
17:            $\mathbf{H}^{(l)} = [ones(1,b); f(\mathbf{A}^{(l)})]$
18:        end for
19:        $\mathbf{A}^{(L)} = (\boldsymbol{\theta}^{(L)})^T\mathbf{H}^{(L-1)}$
20:        $\hat{\mathbf{Y}}_{(t,i)} = [\hat{y}(1),\hat{y}(2),\cdots,\hat{y}(b)] = \mathbf{H}^{(L)} = f_L(\mathbf{A}^{(L)})$
21:        Data selection:
22:        $e(\hat{y}(k),\bar{y}(k)) = (\hat{y}(k)-\bar{y}(k))$, for $k = 1,\cdots,b$
23:        $\mathcal{E} = [e(\hat{y}(1),\bar{y}(1)),\cdots,e(\hat{y}(b),\bar{y}(b))] = \hat{\mathbf{Y}}_{(t,i)} - \mathbf{Y}_{(t,i)}$
24:        $\sigma_e^2 = \text{Var}(\mathcal{E})$
25:        $(\sigma_e^t(i))^2 = (1-\lambda_e)\sigma_e^2 + (\lambda_e)(\sigma_e^t(i-1))^2$
26:        $\mathcal{R} \to \begin{cases} k \notin \mathcal{R}, \text{ if } \frac{|e(\hat{y}(k),\bar{y}(k))|}{(\sigma_e^t(i))} \leq \sqrt{\tau} \\ k \in \mathcal{R}, \text{ otherwise} \end{cases}$, for $k = 1,\cdots,b$
27:        $\mathcal{R} = [k_1,k_2\cdots,k_p]$, $\hat{P}_{\text{up}} = |\mathcal{R}|/b$
28:        $\mathbf{Y}_{\mathcal{R}} = [\bar{y}(k_1),\cdots,\bar{y}(k_p)]$, $\hat{\mathbf{Y}}_{\mathcal{R}} = [\hat{y}(k_1),\cdots,\hat{y}(k_p)]$
29:        Backpropagation:
30:        $\boldsymbol{\Delta}_{\mathcal{R}}^{(L)} = [\boldsymbol{\delta}^{(L)}(k_1),\boldsymbol{\delta}^{(L)}(k_2),\cdots,\boldsymbol{\delta}^{(L)}(k_p)] = f_L'(\mathbf{A}_{\mathcal{R}}^{(L)}) \otimes (\hat{\mathbf{Y}}_{\mathcal{R}} - \mathbf{Y}_{\mathcal{R}})$
31:        for $l = L-1 : -1 : 1$ do
32:            $\boldsymbol{\Delta}_{\mathcal{R}}^{(l)} = f'(\mathbf{A}_{\mathcal{R}}^{(l)}) \otimes [\boldsymbol{\theta}^{(l+1)}\boldsymbol{\Delta}_{\mathcal{R}}^{(l+1)}]_1^{d^{(l)}}$
33:        end for
34:        Updating the weights:
35:        for $l = 1 : L$ do
36:            $\boldsymbol{\theta}^{(l)} = \boldsymbol{\theta}^{(l)} - \frac{\mu}{b\hat{P}_{\text{up}}}\mathbf{H}_{\mathcal{R}}^{(l-1)}(\boldsymbol{\Delta}_{\mathcal{R}}^{(l)})^T$
37:        end for
38:    end for
39:    $\mathcal{L}_{\text{train}}(\boldsymbol{\Theta},\hat{y},y) = \frac{1}{M}\sum_{m=1}^{M}\mathcal{L}_m^1(\boldsymbol{\Theta},\hat{y},y)$ (and $\mathcal{L}_{\text{val}}$ if this set is previously defined)
40: end for
```

By using the same strategy, the total number of flops in the propagation from the second layer to the third layer is $(2j-1)kb$ flops, and from the third layer to the last layer, the matrix multiplication results in $(2k-1)lb$ flops. In the last part, we obtain the estimated signal in NN. In regression problems, the output activation function is linear, resulting in 0 flops. In classification problems, the softmax function has $l-1$ additions, 1 division, and $l+1$ exponential for b training examples, resulting in a total of $(2l+1)b$ flops.

Therefore, the total number of flops for feedforward propagation depends on the problem. For regression problems, it is $(2ij+2jk+2kl-j-k-l)b$ flops, and for classification problems, it is $(2ij+2jk+2kl+l+1)b$ flops.

In backpropagation, the data selection is taken in to account when counting the flops. We compute the sensitivity vector in the fourth layer from the last layer to the third layer. To obtain $\mathbf{\Delta}^{(3)} = \hat{\mathbf{Y}}_{(t,i)} - \mathbf{H}^{(L)}$, it is required $l(P_{\text{up}}b)$ flops. In the remaining layers, we compute the sensitivity weights from the vectors previously obtained. For example, from the third layer to the second layer, we have $\mathbf{\Delta}^{(2)} = f'(\mathbf{A}^{(2)}) \otimes [\mathbf{W}^{(3)}\mathbf{\Delta}^{(3)}]_1^{d(2)}$. The number of flops in the derivative of the activation function is zero. The multiplication matrix $\boldsymbol{\theta}^{(3)}\mathbf{\Delta}^{(3)}$ results in $k(P_{\text{up}}b)$ inner products that involve $l-1$ additions and l multiplications. The element-wise operation has $k(P_{\text{up}}b)$ flops. Then, the resulting number of flops is $2lk(P_{\text{up}}b)$ in this propagation.

Finally, we have the same procedure from the second to the first layer, and the number of flops required is $2kj(P_{\text{up}}b)$.

The last step is the weight updating, for example, from the fourth to the third layer, we have the update $\boldsymbol{\theta}^{(3)} = \boldsymbol{\theta}^{(3)} - \frac{\mu}{b}\mathbf{H}^{(2)}(\mathbf{\Delta}^{(3)})^T$. The multiplication matrix results in a total of $(2(P_{\text{up}}b)-1)kl$ flops, and the sum of the matrices requires kl flops. By summing the multiplications and additions, we end up with $2(P_{\text{up}}b)kl$ flops. The same process is performed in the third to the second layer, $2(P_{\text{up}}b)jk$, and the second to the first layer, $2(P_{\text{up}}b)ij$. Then, the total number of flops in backpropagation is

$$(P_{\text{up}}b)(2ij+2jk+2kl+2kj+2kl+l). \tag{4.27}$$

In backpropagation, the number of flops is a function of the probability of update, reducing the computational cost. Therefore, the total number of flops is organized in Table 4.1.

4.2.4 Simulation results

This section verifies the data selection method's performance in the neural network under different datasets. The proposed method is evaluated, considering regression and classification problems. All algorithms were implemented in MATLAB and are

Table 4.1: Total number of flops in a NN with four layers. The number of nodes of the input layer is i, j is the number of nodes in the second layer, k is the number of nodes in the third layer and l is the number of nodes in the output layer. The number of epochs is n_{ep} (epoch), the number of samples in the mini-batch is b, I is the number of iterations, and P_{up} is the probability of update.

Regression	$(2ij + 2jk + 2kl - j - k - l)b+$ $(P_{\text{up}}b)(2ij + 2jk + 2kl + 2kj + 2kl + l)$
Classification	$(2ij + 2jk + 2kl + l + 1)b+$ $(P_{\text{up}}b)(2ij + 2jk + 2kl + 2kj + 2kl + l)$

available online on GitHub [101]. The simulations were performed on a computer with an Intel Core i7-7500U CPU 2.70GHz x4 processor and 15.5 GB of memory.

As previously mentioned, the algorithm was tested in two combinations of the objective function and the output function: 1) cross entropy error and softmax function for classification problems; 2) mean square error and linear function for regression problems. The activation function chosen for both problems was the ReLU function. The number of layers and units per layer in the hidden layers varies according to the problem. The parameters in this section were established from the leading neural network references followed by this text [88, 89, 102].

The datasets are first preprocessed using the following techniques. The one-hot encoding[1] is used for categorical variables. When selecting the features, a primary criterion adopted is to eliminate a portion of the attributes which are not correlated enough to the output. The last step in data preprocessing consists of normalizing the attributes within the range [0,1], also known as the min-max normalization.

The reduction in the computational cost is the main benefit of the decrease in update probability. The prescribed probability of update $P_{\text{up}} \in \{0.1, 0.3, 0.5, 0.7\}$ is compared with the case where the data is always updated $P_{\text{up}} = 1$.

In the following simulations, the data selection method is applied to the neural network to verify its performance in the testing set and save on computational costs. We solve both regression and classification problems using shallow neural networks in Subsections 4.2.4.1 and 4.2.4.2. In Subsection 4.2.4.3, we consider deep neural networks to tackle regression and classification problems. We also quantify the proposed algorithms' computational complexity by counting the number of flops required to perform the NN, as shown in Subsection 4.2.3, and by the elapsed real-time.

[1]One-hot encoding is a process by which categorical variables are converted into a format so that machine learning algorithms can do better when predicting the classes. For example, suppose the values in a categorical variable range from 0 to $N - 1$. We replace this column with other N columns and apply the following rule: 1 indicates the occurrence of the related category and 0 otherwise [102].

4.2.4.1 Regression Problems

The parameters for the regression simulations are shown in Table 4.2. In the following, we give a brief description of each considered dataset.

Table 4.2: Simulation parameters for the regression problems in shallow NN.

	Problems 1 and 2	Problems 3 and 4
Step size, μ	0.01	0.01
Mini-batch size, b	256	256
Number of epochs, E	200	200
Number of hidden layers	2	2
Number of nodes	128	128
Forgetting factor, λ_e	0.9	0.99

Problem 1: superconductivity dataset

The superconducting material dataset is provided by the University of California at Irvine (UCI) Machine Learning Repository [103, 104]. By using a neural network, a model for the superconducting critical temperature is formulated from features extracted based on thermal conductivity, atomic radius, valence, electron affinity, and atomic mass. A total of 89 preprocessed features are used as input for the neural network. The training and testing sets contain 18,000 and 3,000 samples, respectively.

Table 4.3: Mean squared error obtained in Test (Validation) phase for Problems 1-4 and the average elapsed real time taken to complete one mini-batch iteration for each probability of update P_{up}. Blue represents the best result, whereas red represents the worst result for each problem.

	Problem 1	Problem 2	Problem 3	Problem 4	Time (sec)
$P_{\text{up}} = 1$	0.0086±2.79e-5	6.12e-04±7.52e-6	0.0031±1.06e-5	0.0021±3.91e-5	0.0035
$P_{\text{up}} = 0.7$	0.0082±3.02e-5	4.41e-4±5.89e-6	0.0027±1.8e-4	0.0014±2.74e-5	0.0031
$P_{\text{up}} = 0.5$	0.0079±5.76.e-5	4.12e-04±5.38e-6	0.0029±1.07e-5	0.0012±2.93e-5	0.0028
$P_{\text{up}} = 0.3$	0.0079 ±9.96.e-5	3.20e-4±8.72e-6	0.0024±8.49e-6	0.0010±1.41e-4	0.0025
$P_{\text{up}} = 0.1$	0.0088±3.18e-4	3.56e-04±1.50e-4	0.0023±1.43e-5	0.0009± 2.74e-4	0.0023

Problem 2: online news popularity dataset

The online news popularity dataset comprises 39,644 articles published by Mashable website [105] in two years, and the UCI provides this repository [106]. We estimate a model to predict the popularity of online news by using a neural network. A total of 50 preprocessed features are used as input for the NN. The training and testing sets contain 35,000 and 4,644 samples, respectively.

Problem 3: Facebook comment volume dataset

The UCI machine learning repository provides the Facebook comment volume dataset [107]. We present a neural network model to predict how many comments the Facebook post will receive. This dataset contains 50 preprocessed features extracted from Facebook posts, such as page likes, page category, and publication day. The training and test sets contain 40,988 and 10,120 samples, respectively.

Problem 4: FIFA 19 complete player dataset

The FIFA 19 Complete Player dataset is provided by the Kaggle website [108], which contains all statistics and playing attributes of all players in the version of FIFA 19. A total of 73 preprocessed features are used as input. The training and test sets contain 12,000 and 2,743 samples, respectively.

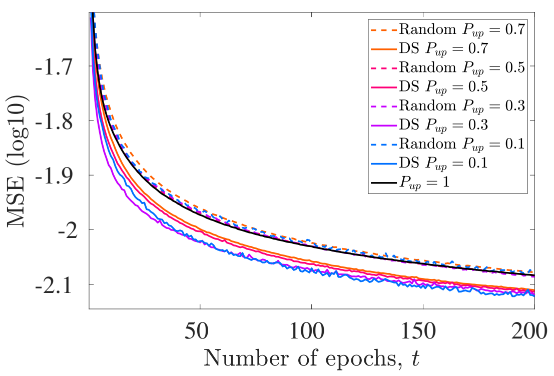
The test mean squared error (MSE) curves are presented as a function of the probability of update P_{up} in Figure 4.2 for Problems 1-4. We may notice an improvement in two-layer NN performance as P_{up} is decreased. Moreover, our data selection outperforms the random selection for all considered probabilities of update.

The average MSE over the last ten epochs is presented in Table 4.3 for regression problems. As shown in Table 4.3, we can achieve optimal results with $P_{\text{up}} = 0.1$ in problems 3 and 4, whereas problems 1 and 2 require $P_{\text{up}} = 0.5$ and $P_{\text{up}} = 0.3$, respectively.

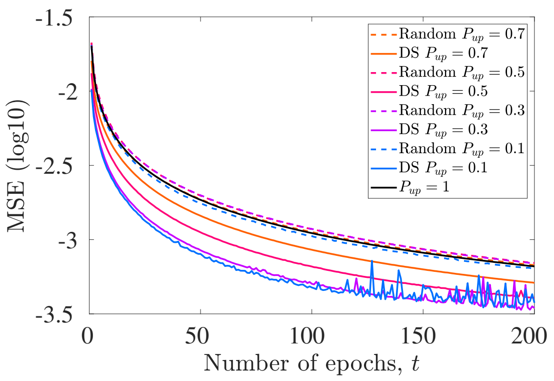
As shown in Figure 4.3, the estimated probability of update $\hat{P}_{\text{up}}$ is quite close to the prescribed probability of update P_{up} for Problems 1-4.

The number of flops required by each regression problem is shown in Table 4.4. Furthermore, to illustrate the complexity reduction, the averaged elapsed real time required to perform one mini-batch iteration is shown in Figure 4.4 when the probability of update is varied.

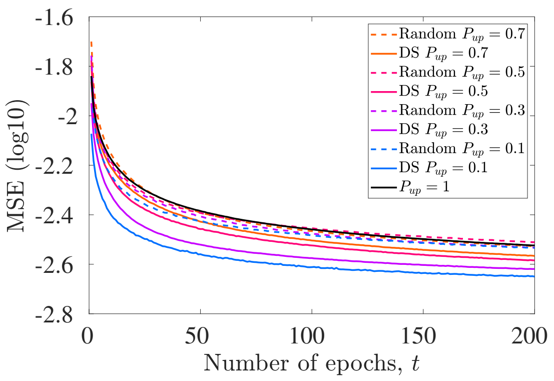
The probability distribution of the error signal is illustrated in Figure 4.5 for the considered regression problems after 100 epochs. We can note that the error distribution is almost normal, reinforcing our hypothesis in equation (4.7).



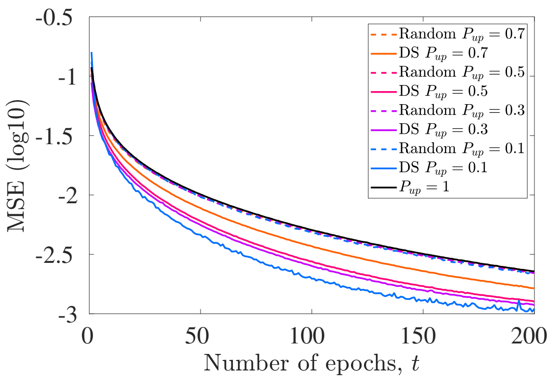
(a) Problem 1: superconductivity dataset



(b) Problem 2: online news popularity dataset



(c) Problem 3: Facebook comment volume dataset



(d) Problem 4: FIFA 19 complete player dataset

Figure 4.2: Test MSE curves comparing the case when the algorithm is always updated $P_{\text{up}} = 1$ with the probability of update $P_{\text{up}} \in \{0.1, 0.3, 0.5, 0.7\}$, for problems 1-4.

Table 4.4: Approximate number of flops in one epoch varying the probability of update P_{up} for Problems 1-4.

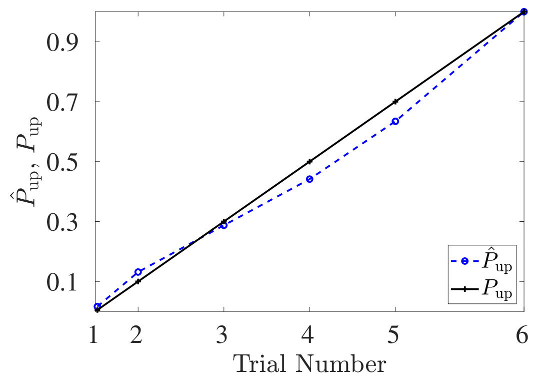
	Problem 1	Problem 2	Problem 3	Problem 4
$P_{\text{up}} = 1$	2599×10^6	43549×10^6	50999×10^6	16349×10^6
$P_{\text{up}} = 0.7$	2119×10^6	35269×10^6	41309×10^6	13299×10^6
$P_{\text{up}} = 0.5$	1799×10^6	2974×10^6	3483×10^6	1125×10^6
$P_{\text{up}} = 0.3$	1479×10^6	2422×10^6	2837×10^6	922×10^6
$P_{\text{up}} = 0.1$	1159×10^6	1870×10^6	2190×10^6	719×10^6

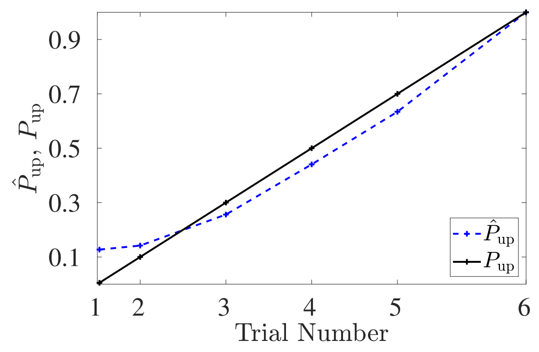
4.2.4.2 Classification problems

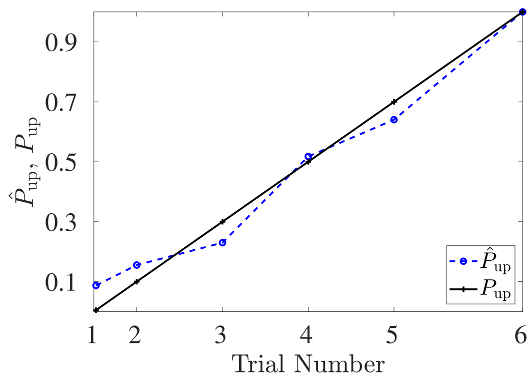
The parameters for the classification simulations are shown in Table 4.5. The datasets considered for classification problems are briefly described as follows.

Problem 5: MNIST Handwritten Digit Recognition Dataset

The Modified National Institute of Standards and Technology (MNIST) [109] database is a large data set containing digits handwritten by students and em-


(a) Problem 1: Superconductivity Dataset


(b) Problem 2: Online News Popularity Dataset


(c) Problem 3: Facebook Comment Volume Dataset

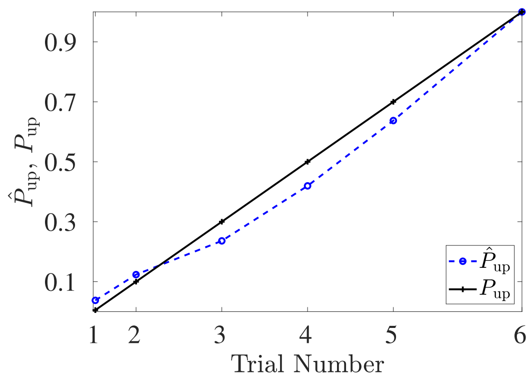

(d) Problem 4: FIFA 19 Complete Player Dataset

Figure 4.3: Comparison between the desired P_{up} and achieved $\hat{P}_{\text{up}}$ for problems 1-4.

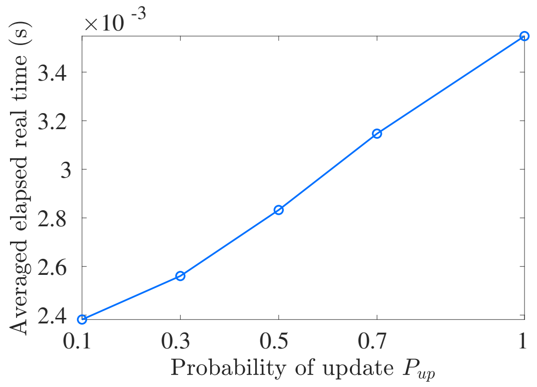
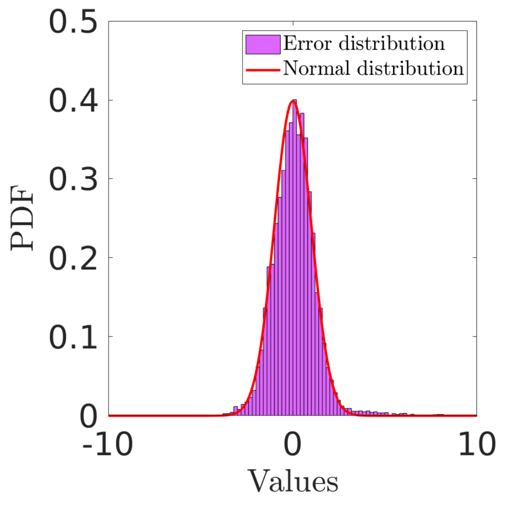
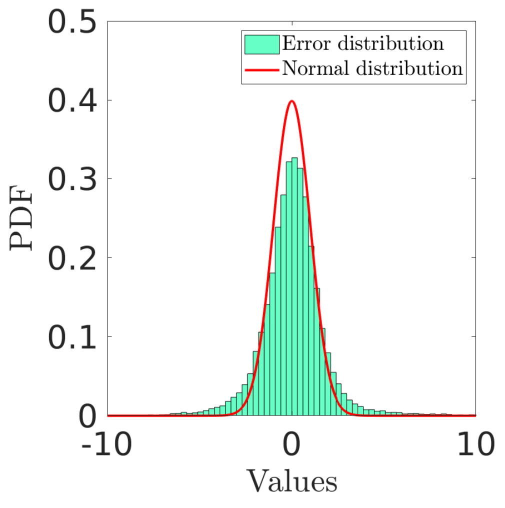


Figure 4.4: Averaged elapsed real time taken to complete one mini-batch iteration as a function of the probability of update P_{up}.

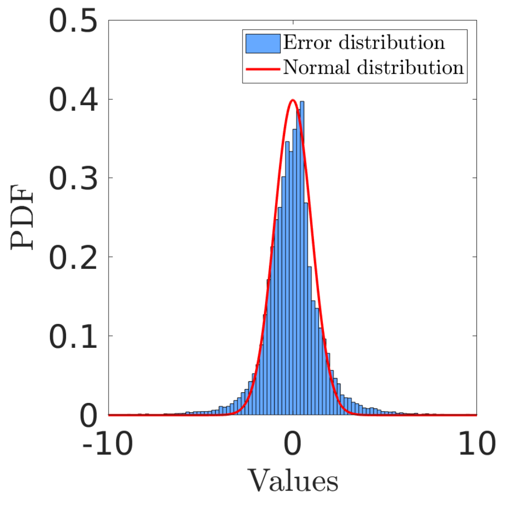
ployees of the United States Census Bureau. The training and testing sets contain 60,000 and 10,000 samples, respectively. This set's input is a 28×28 matrix, where each value represents a pixel of the image. The input signal is normalized in the range 0 to 1. The output labels consist of integer values between 0 and 9.



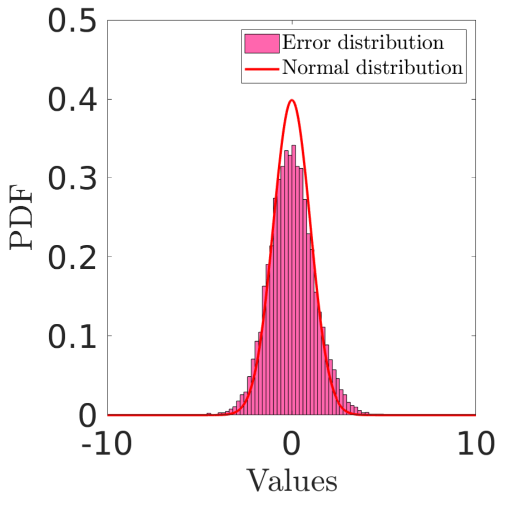
(a) Problem 1: superconductivity dataset



(b) Problem 2: online news popularity dataset



(c) Problem 3: Facebook comment volume dataset



(d) Problem 4: FIFA 19 complete player dataset

Figure 4.5: Probability distribution for the samples in the 100-th epoch for problems 1-4.

Table 4.5: Simulation parameters for the classification problems in shallow NN.

	Problems 5, 6 and 7
Step size, μ	0.01
Mini-batch size, b	256
Number of epochs, E	100
Number of hidden layers	2
Number of nodes	1024

Problem 6: EMNIST letters dataset

The EMNIST letter dataset is provided by the National Institute of Standards and Technology (NIST) [110]. This dataset consists of the 26 letters of the alphabet. The training and testing sets contain 120,000 and 25,000 samples, respectively. The examples have been normalized between 0 and 1. The input is a 28×28 matrix, where each input represents a pixel in the image.

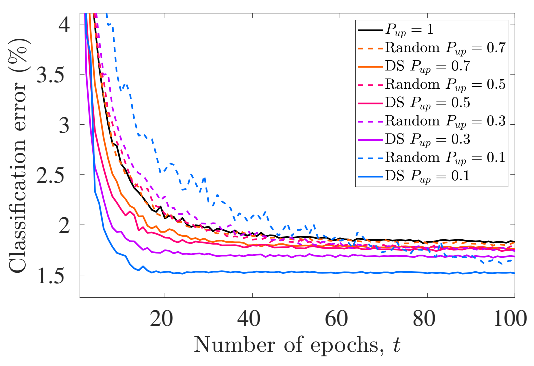
Problem 7: Fashion MNIST dataset

The Fashion-MNIST dataset consists of a training set of 60,000 examples and a test set of 10,000 examples of images of 10 types of clothing, such as shoes, t-shirts, dresses, and more. It is proposed in [111] as a more challenging classification problem than the MNIST. The input of this set is also a matrix 28×28, where each value represents a pixel of the image. The input signal is normalized in the range 0 to 1.

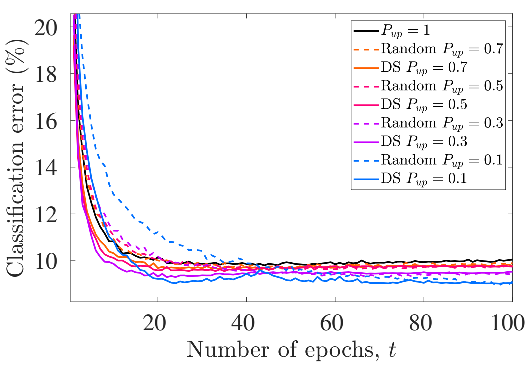
The main results for Problems 5 and 6 are shown in Figure 4.6. Figures 4.6*a* and 4.6*b* depict the results when cross entropy is the objective function and softmax is the output activation function. Again, the data selection method achieves a good performance in the NN, showing that when we decrease the number of updates per epoch, it improves two-layer NN performance ($b = 256$). As in the regression problems, the proposed data selection outperforms the random selection in classification problems. Figure 4.7 illustrates the case in which 2% of the training samples have random labels. Compared with Figure 4.6, we can observe that the overall performance degrades. However, using the data selection method with $P_{\text{up}} \in \{0.1, 0.3\}$, we can obtain low classification error levels.

Figure 4.8 depicts the comparison between the proposed data selection and some related methods. We can observe that the proposed data selection outperforms the online batch selection (OBS) introduced in [112] and the active bias (AB) in [113]. As shown in [114], the OBS method performs slightly worse than the uniform sampling, represented by $P_{\text{up}} = 1$. Instead of selecting a portion of the samples in the minibatch as our method, the OBS method selects the samples to compose the mini-batches. We can also note in Figure 4.8 that our method performs exactly like the online hard example mining method (OHEM) [115] when $P_{\text{up}} = 0.25$. The selection employed in OHEM is the same used in the proposed data selection without the adjusting factor α. The OHEM is designed for a completely different application, as it selects a portion of the examples in the minibatch to create the set of object proposal regions of interest (RoIs).

Table 4.6 presents the averaged MSE for the classification problem's data selec-



(a) Problem 5: MNIST dataset



(b) Problem 6: EMNIST dataset

Figure 4.6: Test classification error (%) comparing $P_{\text{up}} = 1$ with $P_{\text{up}} \in \{0.1, 0.3, 0.5, 0.7\}$, when the output activation function is softmax and objective function is cross entropy error.

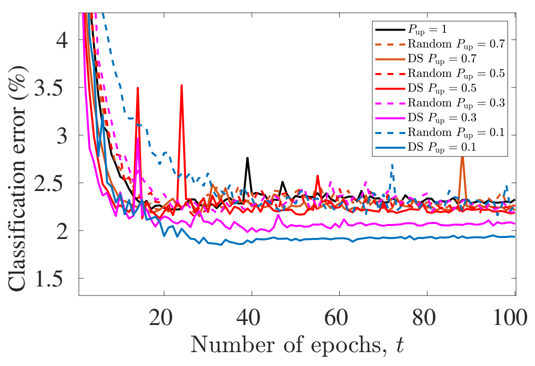


Figure 4.7: Test classification error (%) comparing $P_{\text{up}} = 1$ with $P_{\text{up}} \in \{0.1, 0.3, 0.5, 0.7\}$, when 2% of the training samples have random labels.

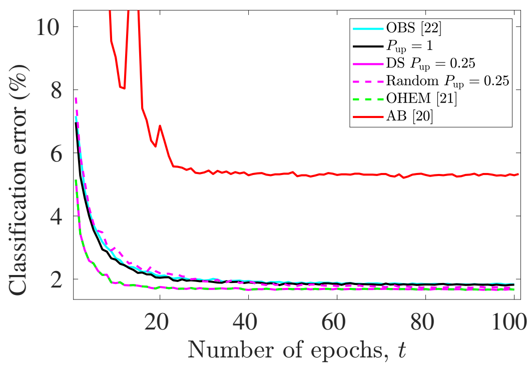


Figure 4.8: Test classification error (%) comparing $P_{\text{up}} = 1$ with $P_{\text{up}} \in \{0.1, 0.3, 0.5, 0.7\}$, and related methods.

tion method over the last ten epochs. In all cases, the method achieves a better result for $P_{\text{up}} = 0.1$, showing that the data selection approach is also suitable for classification problems. Finally, in Table 4.7, we conclude that the data selection reduces the computational cost.

In problem 7, we consider the case where $\alpha = 0.9$; that is, 90% of the elements in set $\mathcal{B}$ are kept for discarding. The remaining 10% are temporarily included in the

Table 4.6: Classification error obtained in Test (validation) phase for Problems 5 and 6. The probability of update P_{up} is varied and compared with no selection case ($P_{\text{up}} = 1$). Blue represents the best result, whereas red represents the worst result for each problem.

	Problem 5	Problem 6
$P_{\text{up}} = 1$	1.87±0.023	9.99±0.069
$P_{\text{up}} = 0.7$	1.78±0.020	9.88±0.056
$P_{\text{up}} = 0.5$	1.72±0.021	9.72±0.059
$P_{\text{up}} = 0.3$	1.69±0.05	9.48±0.042
$P_{\text{up}} = 0.1$	1.50±0.014	9.19±0.034

Table 4.7: Approximate number of flops in one epoch varying the probability of update P_{up} for Problems 5 and 6.

	Problem 5	Problem 6
$P_{\text{up}} = 1$	566×10^9	1145×10^9
$P_{\text{up}} = 0.7$	462×10^9	935×10^9
$P_{\text{up}} = 0.5$	393×10^9	794×10^9
$P_{\text{up}} = 0.3$	324×10^9	654×10^9
$P_{\text{up}} = 0.1$	254×10^9	514×10^9

set $\mathcal{C}$. Then, t_{bin} samples are randomly selected to be in set $\mathcal{C}$. Using an adequate adjustment factor $\alpha < 1$, we can protect the small errors, which are still informative for more complex datasets such as the Fashion MNIST. As illustrated in Figure 4.9, by using $P_{\text{up}} = 0.3$ with $\alpha = 0.9$, we can improve the performance even further. We can also observe that random selection impairs the performance more severely when the dataset is complex, strengthening the use of the proposed data selection strategy.

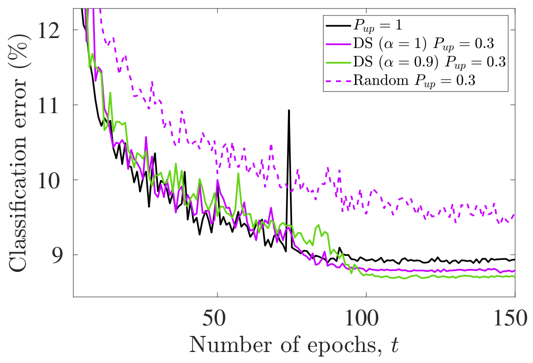


Figure 4.9: Fashion MNIST Dataset. Test MSE curves comparing the case when the algorithm is always updated $P_{\text{up}} = 1$ with the probability of update $P_{\text{up}} \in \{0.3\}$.

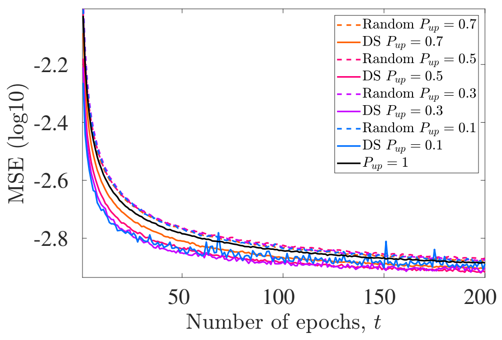


Figure 4.10: Deep neural network simulation: transcoding time dataset. Test MSE curves comparing the case when the algorithm is always updated $P_{\text{up}} = 1$ with the probability of update $P_{\text{up}} \in \{0.1, 0.3, 0.5, 0.7\}$.

4.2.4.3 Deep Neural Network simulations

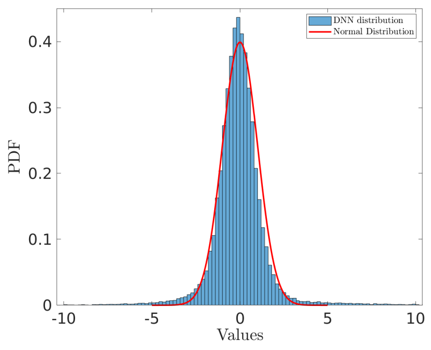
This subsection proposes deep neural network models for regression and classification problems. In the regression simulation, we consider the transcoding time data set, whereas, in the classification simulation, we revisit the MNIST dataset (detailed in Problem 5).

The transcoding time dataset is provided by the UCI Machine Learning Repository [116], and the features include bit rate, frame rate, resolution, and codec, among others. The training and testing sets contain 55,000 and 13,378 samples, respectively. The number of hidden layers is 4, and the number of nodes in each layer is 128. The parameters chosen in this subsection are $\mu = 0.01$, $b = 256$ and $T = 200$. The forgetting factor in the error variance is $\lambda_e = 0.995$.

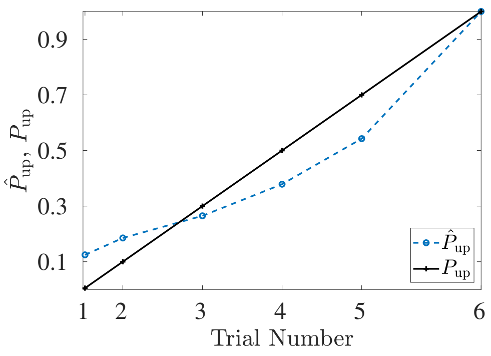
The test MSE curves are illustrated in Figure 4.10 for different probabilities of update P_{up}. We may notice an improvement in the deep neural network performance when $0.1 \leq P_{\text{up}} < 1$. Figure 4.11a shows the probability distribution for all the mini-batches in the 100-th epoch. Again, the distribution is similar to the normal distribution, reinforcing our hypothesis in equation (4.7). Figure 4.11b presents the comparison between the estimated probability $\hat{P}_{\text{up}}$ and the prescribed probability of update P_{up}. We can conclude that the estimated probability is close to the prescribed probability of update.

The next deep learning example verifies the performance of the data selection method in a classification problem. In this case, we consider the MNIST dataset, which was previously detailed in Problem 5. The framework parameters are defined as: the number of hidden layers equals 3, and the number of nodes in each layer is 1024. And the other parameters are chosen as $\mu = 0.1$, $b = 128$ and $E = 100$. The results of this deep learning example are illustrated in Figure 4.12.

To corroborate the suitability of the proposed data-selection method to be applied to alternative architectures, Figure 4.13 shows the performance of a convolutional neural network (CNN) in a classification problem. The curves show the clas-



(a) Probability distribution for the samples selected in the 100-th epoch



(b) Comparison between the desired P_{up} and achieved $\hat{P}_{\mathrm{up}}$

Figure 4.11: Deep neural network simulation: transcoding time dataset.

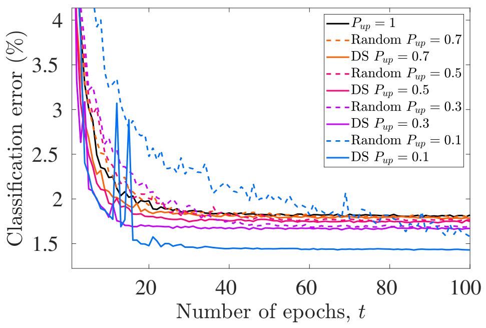


Figure 4.12: Deep neural network simulation: MNIST handwritten digit recognition dataset. Test classification error (%) comparing the case when the algorithm is always updated $P_{\mathrm{up}} = 1$ with the probability of update $P_{\mathrm{up}} \in \{0.1, 0.3, 0.5, 0.7\}$.

Table 4.8: Approximate number of flops in one epoch for MNIST dataset and transcoding time dataset problems.The probability of update P_{up} is varied $P_{\mathrm{up}} \in \{0.1,0.3,0.5,0.7,1\}$.

	MNIST dataset	Transcoding time dataset
$P_{\mathrm{up}} = 1$	944×10^9	16966×10^6
$P_{\mathrm{up}} = 0.7$	764×10^9	13604×10^6
$P_{\mathrm{up}} = 0.5$	645×10^9	11362×10^6
$P_{\mathrm{up}} = 0.3$	525×10^9	9120×10^6
$P_{\mathrm{up}} = 0.1$	405×10^9	6879×10^6

sification error versus the epochs for the CIFAR10 dataset employing the Resnet18 network model. When compared with the case that all samples are selected ($P_{\mathrm{up}} = 1$), we can achieve a lower classification error with $P_{\mathrm{up}} = 0.5$ in the last epochs.

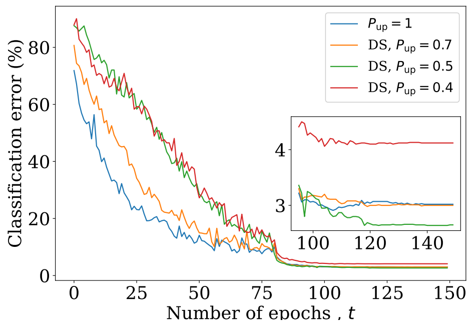


Figure 4.13: Test classification error (%) comparing $P_{\text{up}} = 1$ with $P_{\text{up}} \in \{0.4, 0.5, 0.7\}$ for Resnet18 model and CIFAR10 dataset.

4.3 Adversarial training with informed data selection

Generating adversarial examples during training can be highly computationally intense since each sample is usually built with several steps in the direction of the gradient as the model is trained. Moreover, adversarial training generally decreases the *standard accuracy*, that is, the accuracy on clean samples [117]. This *robustness-accuracy* tradeoff is reported to be highly data-dependent, especially regarding the data distribution [118] and its quality [119]. Furthermore, we only have access to a training dataset that does not necessarily represent the problem we aim to learn. In this case, we could avoid using the entire training data. Since the dataset is reduced, we can save several computations during backpropagation and speed-up training. This hypothesis was already investigated for standard training in [18, 19]. In this work, we extend the work in [18, 19] and apply it to the adversarial training case. The proposed data selection algorithm selects the most relevant samples based on the cross entropy loss from each mini-batch composed of both clean and adversarial samples. Since only the selected samples are used to update the model parameters in the backpropagation, the training time is reduced. The selection also balances the necessary amount of clean and adversarial samples to yield satisfactory robustness and standard accuracy.

When performing adversarial training, we are interested in learning a process or function $f(\cdot)$ that maps a data space $\mathcal{X}$ into an output space $\mathcal{Y}$. However, we do not have direct access to samples from $\mathcal{X}$ to train the model according to the adversarial objective. Instead, we only have access to a subset $\mathcal{D}$ which is split into batches used to update the model parameters in equation (6.22). However, there is no guarantee that this available subset or its batches consist of a good representation of the process $f(\cdot)$. Therefore, we propose a sampling strategy to select the most relevant samples to compose the batches in adversarial training.

4.3.1 Data selection procedure

We first consider the entire original dataset $\mathcal{D}$ of input-output pairs in equation (3.1). Then, at each mini-batch iteration, b' clean samples are selected from the whole dataset to form the batch set $\mathcal{B}'$. By using PGD, b' adversarial examples are generated from the samples in the set $\mathcal{B}'$ using equation (3.16). The resulting mini-batch $\mathcal{B}$ is then composed of $b = 2b'$ samples. The samples in the mini-batch flow through the network, the gradients are computed, and we obtain the network output as a one-hot-encoded vector $\mathbf{y}$, as shown in Figure 4.14. In order to quantify the relevance of the samples in the mini-batch, we define the error signal

$$E(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{c=1}^{C} e(\hat{y}_c, y_c), \tag{4.28}$$

based on the cross entropy loss defined in equation (4.20), in which $C = d^{(L)}$ is the number of classes.

As a rule, the closer to zero the error signal is, the less informative or relevant will be the contribution of the correspondent data pair to the parameter update in equation (3.3) of Chapter 3, and it is repeated as follows for the reader convenience

$$\mathbf{\Theta}_{k+1} = \mathbf{\Theta}_k - \mu \frac{1}{b} \sum_{\mathbf{x}, \mathbf{y} \in \mathcal{B}} \nabla_{\mathbf{\Theta}} \mathcal{L}(f_{\mathbf{\Theta}}(\mathbf{x}), \mathbf{y}), \tag{4.29}$$

where μ is the step size and b is the mini-batch size.

We then propose to select a portion P_{up} of the samples in $\mathcal{B}$ based on the higher error values in equation (4.28), forming a selection set $\mathcal{S}$. After the forward propagation is completed, only the samples in $\mathcal{S}$ are used in the backpropagation to update the network parameters $\mathbf{\Theta}$, as depicted in Figure 4.15. Since only a portion P_{up} of the samples are used to update the parameters, we can save some computations and alleviate the training burden.

One question remains about choosing an adequate P_{up} for our problem. As $P_{\text{up}} \to 0$, fewer samples are selected, and we save more computations in the backpropagation. In this case, however, the selected samples might be insufficient to learn the problem. For standard training, the most favorable P_{up} choice mainly depends on the dataset complexity [19]. Simpler datasets like MINIST require $P_{\text{up}} = 0.3$, whereas for more complex datasets such as CIFAR10, $P_{\text{up}} = 0.5$ is a better choice. Thus, one option is to set a fixed P_{up} for the whole training process. This way, we can set the number of saved computations from the beginning. Nevertheless, in cases where the dataset complexity is unknown and it is difficult to prescribe a P_{up} for all the epochs, an automatic P_{up} can be advantageous. In this way, we can obtain the P_{up} for each epoch in an adaptive manner as the training is performed.

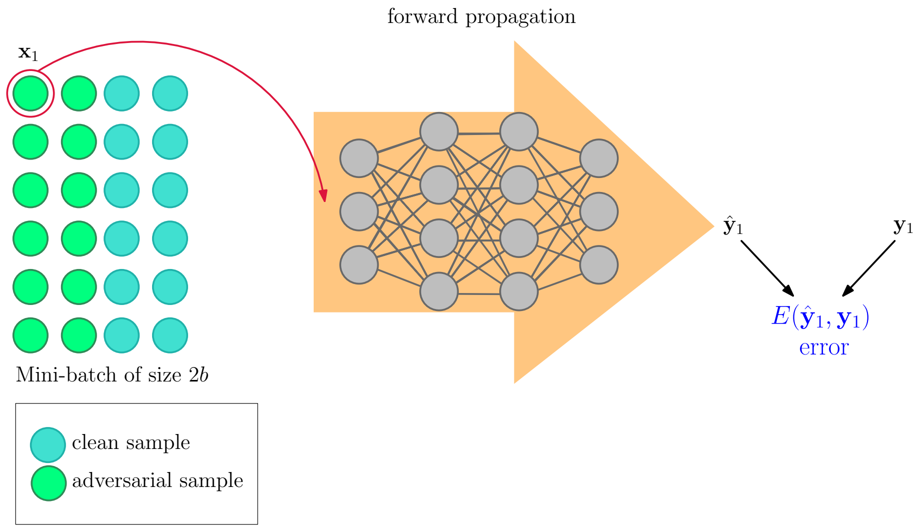


Figure 4.14: Forward propagation and error signal computation.

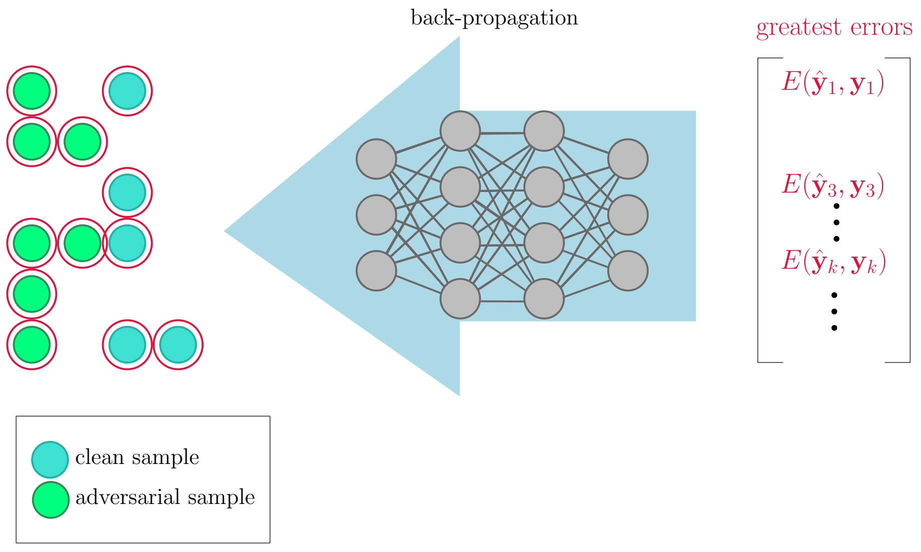

back-propagation
greatest errors
$E(\hat{\mathbf{y}}_1, \mathbf{y}_1)$
$E(\hat{\mathbf{y}}_3, \mathbf{y}_3)$
$E(\hat{\mathbf{y}}_k, \mathbf{y}_k)$
clean sample
adversarial sample

Figure 4.15: Selected samples being used in the backpropagation.

The adaptive procedure considers the accuracy

$$\lambda_{acc} = \frac{N_c}{2b}, \tag{4.30}$$

at each epoch as a criterion, in which N_c is the number of correctly predicted samples and $2b$ is the total number of data points in the batch. Hence, we can estimate the number of selected samples P_{up} at each epoch t.

$$P_{\text{up}}^{(t)} = (1 - \lambda_{acc}^{(t-1)})P_{\text{up}}^{(t-1)} \tag{4.31}$$

where $P_{\text{up}}^{(0)} = 1$ and $\lambda_{acc}^{(t-1)}$ is the last available accuracy. We need more samples in the mini-batch to improve learning when the accuracy is low, whereas fewer samples are required to continue learning when accuracy increases.

As it will be shown in the simulations, updating the P_{up} using equation (9) accelerates the convergence for $P_{\text{up}}^{(0)} = 1$ because, in this case, it selects more samples in the first epochs. Our motivation was to provide more samples to the model at the beginning to improve and accelerate its learning. Therefore, early stopping methods [120] can be employed to reduce the training time. Since we do not consider the early stopping approach in the simulations, we propose using a fixed prescribed P_{up} in this work. The main proposed algorithm is detailed in Algorithm 7.

Algorithm 7 Proposed data selection for adversarial training

```
1:  Given dataset $\mathcal{D}$, mini-batch size $b'$, and prescribed $P_{\text{up}}$
2:  for epoch = $1 \cdots T$ do
3:      for mini-batch $\mathcal{B} \subset \mathcal{D}$ do
4:          Create adversarial examples $\{x'_1, \cdots x'_{b'}\}$ from clean samples $\{x_1, \cdots x_{b'}\}$
5:          using the current state of the network and obtaining
6:          $\mathcal{B}' = \{x'_1, \cdots x'_{b'}, x_1, \cdots x_{b'}\}$;
7:          Forward propagation with samples in $\mathcal{B}'$;
8:          Compute the error signal for each sample in $\mathcal{B}'$ using equation (4.28);
9:          Select the $P_{\text{up}} \times 100\%$ of the samples in $\mathcal{B}'$ with the greatest error values;
10:         Update model parameters by backpropagation using only the data
11:         samples in $\mathcal{S}$;
12:     end for
13: end for
```

4.3.2 Simulation results

In this subsection, we assess the performance of the proposed data selection method in the CIFAR10 dataset [121] using the Resnet18 model [122]. To build the adversarial examples, we consider the PGD attack with $\epsilon = 8/255$, $\alpha = 0.01$, and 20 iterations. We consider the following methods in the simulations. The standard method trains only with clean samples with a mini-batch $\mathcal{B}$ of size $b = 256$. Also, the robust method trains only with adversarial samples with a mini-batch $\mathcal{B}$ of size $b = 256$. The DS robust method is trained with the selection set $\mathcal{S}$ of size $b = 256$, which is composed of both clean and adversarial samples, and it is obtained using our selection strategy with P_{up} fixed or varying. The random robust method is trained with a mini-batch of size $b = 256$, composed of clean and adversarial samples selected randomly. We also consider the selection method proposed in [119], in which the samples are selected based on their learning stability. In this case, we used 50% of the samples with high quality to perform a fair comparison in terms of the number of samples used.

In order to determine the impact on the standard and robustness accuracy at the last epoch, we first vary the proportion of the selected samples in Figure 4.16. We slightly outperform the method that takes into account all samples by utilizing

$P_{\text{up}} = 0.5$, with the advantage of only requiring 50% of the batch's samples. When it comes to robustness, techniques with $0.5 \leq P_{\text{up}} < 1$ perform very similarly to those with $P_{\text{up}} = 1$. Further reducing P_{up} does not result in a performance improvement. In such a scenario, the model would need either more samples or more epochs to understand the problem.

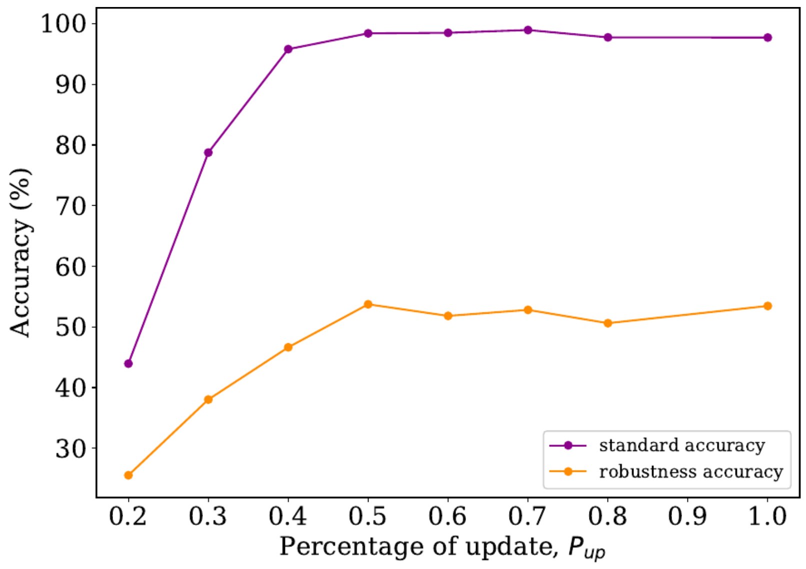


Figure 4.16: Evolution of the standard and robustness accuracy as the portion of selected samples P_{up} is varied.

We then evaluate the proposed DS robust method with varying P_{up} and compare it with the fixed $P_{\text{up}} = 0.5$, the standard and robust methods in terms of standard and robust accuracy in Figures 4.17 and 4.18. Standard accuracy is the accuracy on clean samples, whereas robust accuracy is the accuracy on unseen adversarial examples. We show in Figure 4.19 the obtained P_{up} for each epoch following equation (4.31). By using both a varying P_{up} and $P_{\text{up}} = 0.5$, we observe an improvement in terms of standard accuracy when compared with the standard and robust methods. Moreover, reducing the number of samples in the mini-batch does not affect the robust accuracy, as shown in Figure 4.18.

This improvement in robustness-accuracy tradeoff is reasonable since our method includes the most potentially relevant clean and adversarial samples in the mini-batch. Some claim that such a tradeoff exists because the standard and robust objectives conflict [123, 124]. We can then observe in Figure 4.20 that the model trained with $P_{\text{up}} = 0.5$ starts by selecting more adversarial samples than clean samples. However, after a few epochs, this behavior changes, and the number of selected clean samples increases. This feature potentially suggests that the model tries to learn the adversarial problem first. When done, the DS method attempts to improve the clean accuracy. Moreover, the number of selected minimum adversarial examples increases as the model is trained, as depicted in Figure 4.21. The minimum

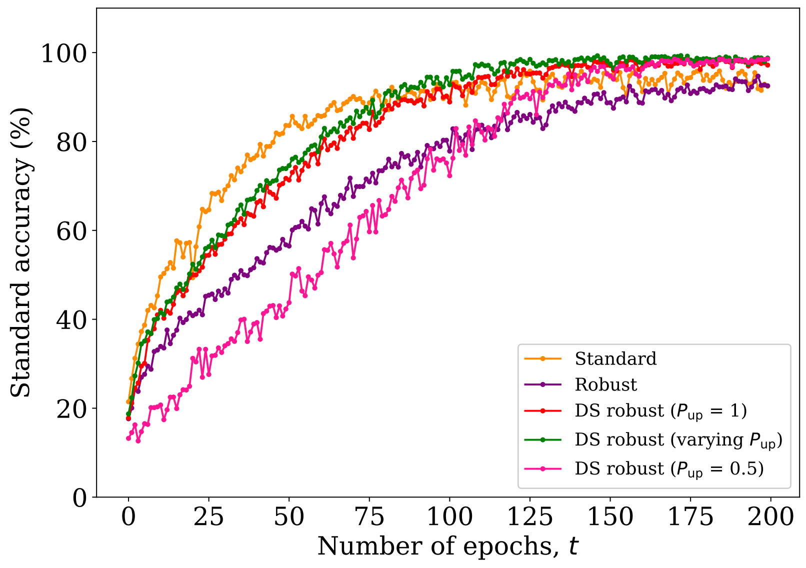


Figure 4.17: Standard accuracy as a function of the number of epochs.

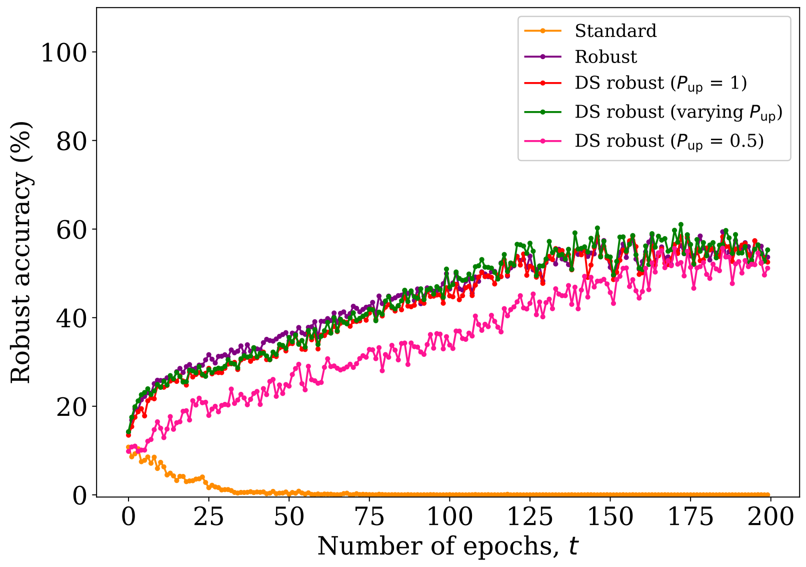


Figure 4.18: Robust accuracy as a function of the number of epochs.

adversarial examples are generated by slowly increasing the perturbation constraint ϵ until the prediction changes.

Finally, our methods are compared with other selection methods in terms of standard and robust accuracy in Figures 4.22 and 4.23, respectively. The DS approach outperforms both the random and selection methods with 50% of high quality samples from [119], especially in terms of standard accuracy.

The benefits of the proposed methods in terms of performance are followed by a reduction in computational complexity. Since only P_{up} samples in the mini-batch are backpropagated through the network to update its parameters, we can save some computations. For example, we present the total training time after 200 epochs in Table 4.9. The simulations were performed on a computer with two GTX-1080

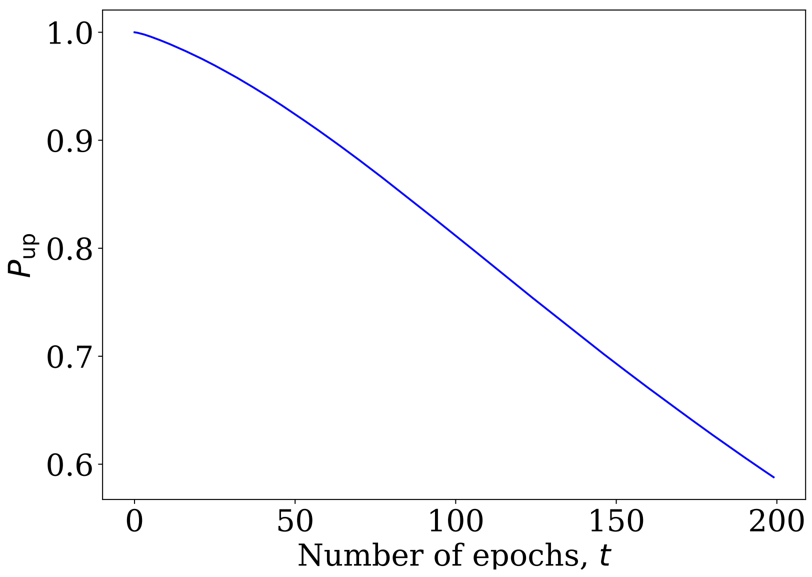


Figure 4.19: Portion of selected samples P_{up} obtained for each epoch following equation (4.31).

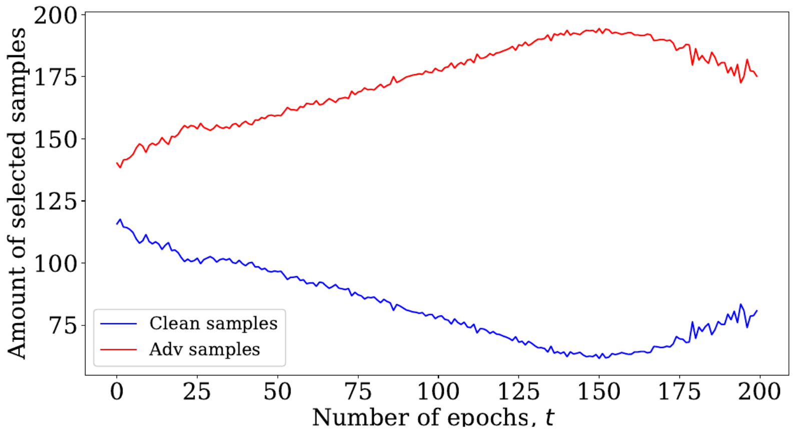


Figure 4.20: Averaged amount of selected clean and adversarial samples at each epoch for $P_{\text{up}} = 0.5$.

GPUs. With $P_{\text{up}} = 0.5$, the training time is reduced when compared with $P_{\text{up}} = 1$ and varying P_{up}. However, if we stop the training by the 150th epoch, the training time for the varying P_{up} can be reduced to 15261.29s. Therefore, the varying P_{up} strategy can be applied if an early stopping method is employed. We also outperform the method introduced in [119] in total training time as their method needs a pre-training to rank the samples by the learning stability values.

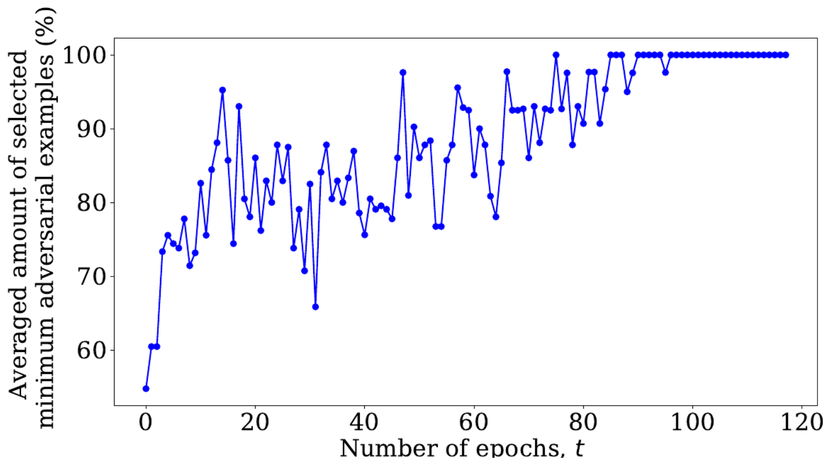


Figure 4.21: Averaged amount of selected minimum adversarial examples at each epoch for $P_{\text{up}} = 0.5$.

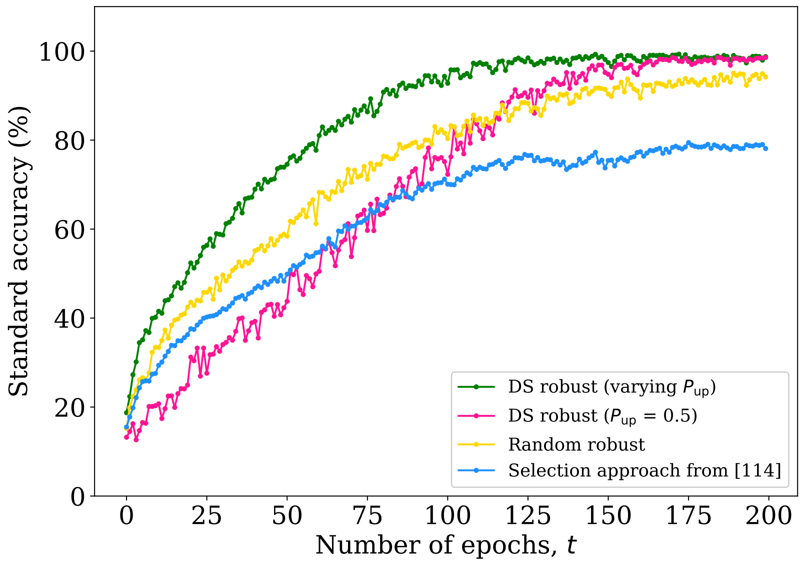


Figure 4.22: Comparing the proposed method with other selection methods in terms of standard accuracy.

Table 4.9: Total training time after 200 epochs.

Method	Time (s)
Selection approach from [119] with 50% of samples removed	39970.71
Robust with $P_{\text{up}} = 1$	20200.33
DS Robust with $P_{\text{up}} = 0.5$	19770.51
DS Robust with P_{up} varying as in equation (4.31)	20161.29

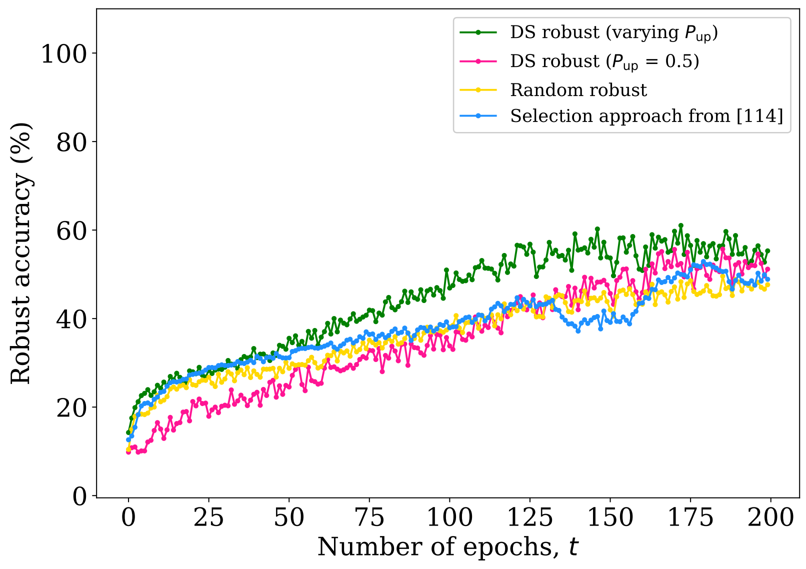


Figure 4.23: Comparing the proposed method with other selection methods in terms of robust accuracy.

4.4 Conclusions

This chapter presented a data selection strategy applied to both standard and adversarial training.

The standard training approaches for neural and deep neural networks do not perform data selection, even if data does not bring a novelty to the learning process. The proposed method was applied in several datasets for classification and regression problems. In all simulations, the data selection achieves excellent performance in terms of MSE and classification error even when only 30% of the data is utilized, proving an excellent tool to improve the training process. Therefore, data selection in NN is a helpful tool, mainly with the advantage of requiring a lower computational cost than standard methods. Besides, it enables the prescription of how often to update the weights of the neural network.

Adversarial training is the most popular solution to mitigate the effect of malicious attacks on deep neural networks. However, although adversarial training can improve the robustness accuracy, it usually sacrifices standard accuracy. Motivated by this drawback and seeking to reduce the computational complexity during training, we proposed a data selection strategy to include the data samples that bring novelty to the learning process. The simulation results with CIFAR10 using the Resnet18 model indicate that the method is beneficial in improving the robustness-accuracy tradeoff and reducing the computational complexity of the training.

In the following chapter, we return to the core of this work and describe the considered OFDM systems, their challenges, and our proposed ML solutions.

Chapter 5

Machine learning for wireless OFDM systems

5.1 Introduction

The machine learning techniques were presented in Chapter 3, and a data selection strategy was detailed in Chapter 4. In this chapter, we employ the ML techniques presented in Chapter 3 to solve tasks regarding practical problems in wireless OFDM systems. The method for selecting data from Chapter 4 is not used in the experiments of this chapter, but it is simple to use them in future research.

In broadband communications, the available spectrum should be utilized to maximize the data transmission while reducing the amount of information needed to deal with channel distortions. The most widely used solution to address multipath fading in the wireless communications field is the orthogonal frequency-division multiplexing (OFDM) modulation technique due to its simplicity and effectiveness while requiring some extra bandwidth. OFDM systems combat inter-symbol interference (ISI) and inter-block interference (IBI) caused by the multiple delayed versions of the transmitted signal [14], utilizing L redundant elements as prefix or sufix. L depends on the maximum delay spread of the channel [15], and the crucial role of the redundant elements is to avoid IBI. The redundancy consists of a cyclic prefix (CP) or zero padding (ZP) as sufix, giving rise to distinct types of OFDM transceivers as well as single carrier solutions with frequency-domain equalization; see [15] for details.

In many applications, the loss in spectrum efficiency might be an issue, so OFDM systems with reduced or free of redundancy become attractive alternatives [125], as long as the additional complexity related to the transceiver implementation is viable. A particular case is ZP-OFDM, allowing a zero-forcing solution requiring the minimum amount of redundancy of $L/2$ for even L [126, 127]. Unfortunately this solution is prone to face numerical difficulties. In contrast, a single redundant element is necessary if some specific types of time-varying transceivers are designed [128]. This work discusses alternative solutions to devise block transceivers, starting from free to full-length prefix.

Many machine learning tools are currently being utilized in many distinct applications. In particular, machine learning algorithms have disrupted the fields of natural language processing and computer vision. This enormous success of ML has sparkled the interest in investigating the application of deep neural networks (DNNs) in wireless communications [125, 129–132]. The choice of deep models represents the belief that the communication system we want to learn can be represented through a composition of several simpler systems [133]. Indeed, the receiver of an OFDM system consists of several blocks, which can be modeled by DNNs [129]. Moreover, in some related works, it is conjectured that more control can be obtained by exploiting the expert knowledge in wireless communications and breaking, for example, a single DNN in two [125, 130].

Currently available communications systems solutions are mature because it is possible to design the most appropriate transmission and reception solution for many well-understood and mathematically modeled physical layer environments. The design entails choosing, for instance, the appropriate components such as symbol constellation, modulation method, the family of typical channel models, environment noise properties, amount and format of the training signals, etc. For most standardized communications services, components are well defined, and the deployed systems work well as long as the pre-established assumptions related to the physical layer are met, e.g., the channel model is linear, or the environment noise is additive white Gaussian noise (AWGN). In these cases, the available transceiver solutions meet high-performance standards that are not necessarily surpassed by replacing them with machine learning (ML) based solutions. However, in environments where assumptions depart from reality or no available solutions are using well-established communications tools, the solutions incorporating ML might bring about tremendous benefits. Whenever there is a gap between the available theoretical models and those faced in practice, ML-based solutions are expected to provide better answers.

In recent years, many works have investigated strategies for applying the concept of ML to solve problems related to performance, robustness, and detection in communication systems, particularly those aspects related to the physical layer [134–137]. Although many tasks in the physical layer of communication systems are already optimally solved, significant practical problems still require acceptable solutions [138].

We start, in Section 5.2, by describing the OFDM system models considered in this work. Then, the basic simulation setup utilized in this chapter and the next one is presented in Section 5.3. Among the practical problems in OFDM systems, we can mention the spectrum waste and nonlinear distortion caused by techniques to reduce the peak-to-average power ratio (PAPR). For this problem, we propose a two-block ML-based OFDM reception operating with insufficient redundancy in Section 5.4. Our work answers how to perform a solution lying between a prefix-free and full prefix OFDM. The first block is responsible for inverting an unknown function: the channel and the nonlinear distortion. The second one aims to detect the transmitted OFDM symbol corrupted by ISI and IBI. We consider both CP-transceivers and ZP-transceivers such as ZP-OFDM-OLA (zero padding-OFDM-overlap and add) and ZPZJ (zero padding zero jamming) OFDM [15]. This solution comprises our contributions in [21, 22]. The proposed approach is also suitable for OFDM systems with the comb-type pilot arrangement, where fewer pilots can be transmitted than in the block-type case. Furthermore, the comb-type pilot arrangement is more favorable for fast-fading channel scenarios [139, 140]. This solution is proposed in Subsection 5.4.3 and consists of our submitted work [23].

As the communication environment is constantly changing, adaptive mechanisms for switching to a reinforcement learning mechanism [141] to better manage the system resources are highly desirable. For such online tasks, we propose a deep reinforcement learning approach to choose the amount of redundancy the transmitter should employ to satisfy an MSE target performance in Section 5.5.

Finally, we end the chapter with some concluding remarks in Section 5.6.

5.2 OFDM system model

Consider the OFDM transmitter illustrated in Figure 5.1. The orange blocks represent the ML methods attached to the main OFDM system. We propose standard trained neural networks in this chapter, whereas the robust version is presented in the next chapter. The incoming data streams are modulated by using the M-ary quadrature amplitude modulation (M-QAM), resulting in the data symbols $\boldsymbol{x}_D^{\mathrm{T}} \in \mathbb{C}^{1\times N_D}$. In the subcarrier allocation block, the pilot symbols $\boldsymbol{x}_P^{\mathrm{T}} \in \mathbb{C}^{1\times N_P}$ are inserted to form the symbols $\boldsymbol{x} \in \mathbb{C}^{N\times 1}$. We consider the pilot arrangements depicted in Figure 5.2. For the block-type arrangement, shown in Figure 5.2a, the symbols are defined as

$$\boldsymbol{x}^{\mathrm{T}} = \boldsymbol{x}_P^{\mathrm{T}} \ \text{ or } \ \boldsymbol{x}_D^{\mathrm{T}}, \tag{5.1}$$

so that all subcarriers either contain pilots or data. Besides, in the comb-type arrangement, shown in Figure 5.2b, the symbols are defined as

$$x(n) = \begin{cases} x_P(n), & \text{for} \quad n \in \mathbb{B}_P \\ x_D(n), & \text{for} \quad n \in \mathbb{B}_D, \end{cases} \tag{5.2}$$

where sets $\mathbb{B}_P$ and $\mathbb{B}_D$ contain the indices of the subcarriers allocated to carry N_P pilots and N_D data symbols, respectively. The subcarriers can carry pilot and data symbols in the comb-type pilot arrangement.

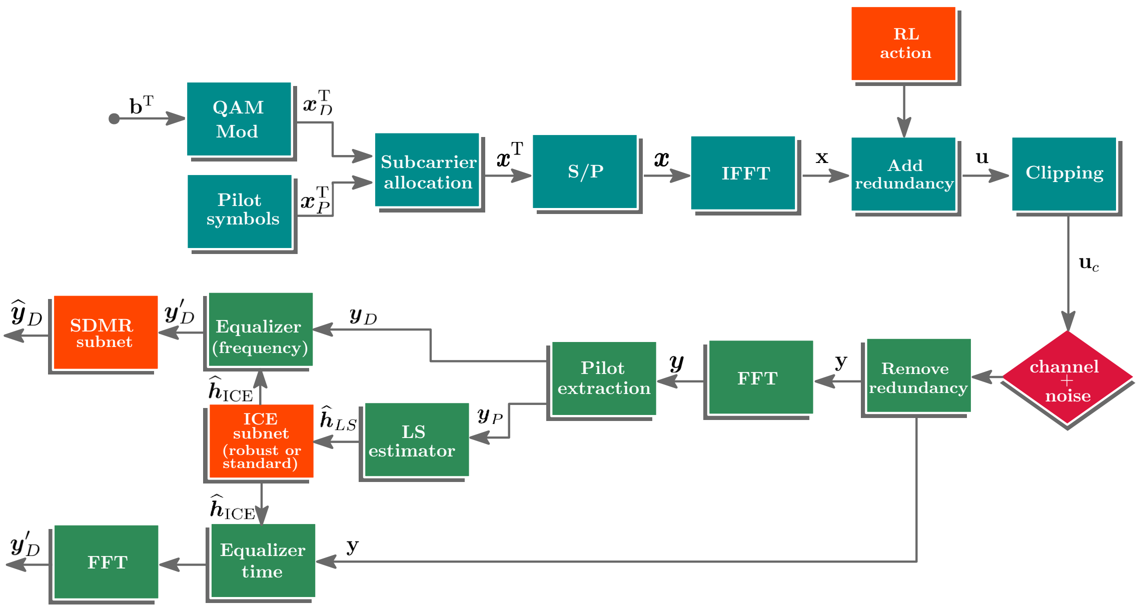


Figure 5.1: Proposed OFDM transceivers.

The symbols are then converted to a parallel data stream $\boldsymbol{x} \in \mathbb{C}^{N\times 1}$. The N-point inverse fast Fourier transform (IFFT) is employed to convert the signal from the frequency domain $\boldsymbol{x}$ to the time domain $\mathbf{x}$. The K-length redundancy is added, resulting in the OFDM signal

$$\mathbf{u} = \mathbf{A}\mathbf{W}_N^{\mathrm{H}}\boldsymbol{x} = \mathbf{A}\mathbf{x}, \tag{5.3}$$

where $\mathbf{W}_N$ is the unitary $N \times N$ discrete Fourier transform (DFT) matrix. Matrix $\mathbf{A} \in \mathbb{C}^{S\times N}$ adds redundancy in which

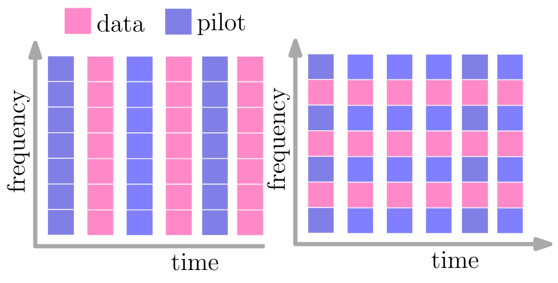
$$\mathbf{A} = \begin{cases} \mathbf{A}_{\mathrm{ZP}} = \begin{bmatrix} \mathbf{I}_N \\ \mathbf{0}_{K\times N} \end{bmatrix}, \text{ for zero-padding} \\ \mathbf{A}_{\mathrm{CP}} = \begin{bmatrix} \mathbf{0}_{K\times(N-K)} \ \mathbf{I}_K \\ \mathbf{I}_N \end{bmatrix}, \text{ for cyclic-prefix.} \end{cases} \tag{5.4}$$

One of the drawbacks of OFDM systems is the high peak-to-average power ratio (PAPR) due to the addition of the many subcarrier components via IDFT operation. The main problem is that the gain of the transmitter power amplifier will saturate at high input power. Therefore, the power amplifier would have to operate in a back-off regime, but it requires an excess large saturation power for the power amplifier. The clipping technique is a simple approach to reduce PAPR at the transmitter side and avoid operating the amplifiers in their linear region. Nevertheless, clipping is a nonlinear process that might degrade the bit-error-rate performance [142]. The clipped version of $\mathbf{u}$ is expressed as

$$\mathbf{u}_c = \begin{cases} \mathbf{u}, & \text{if } |\mathbf{u}| < (C_R \sigma_u). \\ \frac{\mathbf{u}}{|\mathbf{u}|}(C_R \sigma_u), & \text{otherwise,} \end{cases} \tag{5.5}$$

where C_R is the clipping ratio, and σ_u is the root mean square (RMS) value of the OFDM signal [142]. Low clipping ratio C_R leads to more severe nonlinear distortion, whereas $C_R \to \infty$ represents the absence of clipping noise. Then N subcarriers are used to transmit the OFDM symbol $\mathbf{u}_c \in \mathbb{C}^S$, with $S = N + K$.

As in [125, 130], we assume that the channel remains approximately constant over the transmission of an OFDM frame. We also consider that only neighborhood blocks interfere with each other. In the block-type arrangement depicted in Figure 5.2a, the pilot signal is transmitted to obtain an estimate for the channel impulse response, $\widehat{\mathbf{h}}$ at each subcarrier. This configuration is also considered in [125, 130] to simplify the simulations.



(a) Block-type (b) Comb-type

Figure 5.2: Pilot arrangements.

The data signal is then transmitted using all the N subcarriers, and the previously obtained channel estimate is used to estimate the transmitted symbols via equalization.

The comb-type arrangement shown in Figure 5.2b allows the subcarriers to carry both pilot and data symbols. In this case, the channel estimate is obtained at the pilot subcarriers. The channel estimate at the data subcarriers for comb-type pilot arrangement is obtained via interpolation. Some examples of interpolation methods are linear interpolation, second-order polynomial interpolation, and cubic spline interpolation. The transmitted data symbols are then estimated after channel equalization.

5.2.1 Channel estimation

The channel model between transmitter and receiver has the impulse response $\mathbf{h} = [\mathrm{h}(0) \;\; \mathrm{h}(1) \;\; \cdots \mathrm{h}(\mathrm{L})]^{\mathrm{T}}$. In the z-domain, the pseudo-circulant channel matrix is

$$\begin{aligned}\mathbf{H}(z) &= \begin{bmatrix} \mathrm{h}(0) & 0 & 0 & \cdots & 0 \\ \mathrm{h}(1) & \mathrm{h}(0) & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathrm{h}(\mathrm{L}) & \mathrm{h}(\mathrm{L}-1) & \ddots & \cdots & 0 \\ 0 & \mathrm{h}(\mathrm{L}) & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \mathrm{h}(\mathrm{L}) & \cdots & \mathrm{h}(0) \end{bmatrix}_{S\times S} \\ &+ z^{-1} \begin{bmatrix} 0 & \cdots & 0 & \mathrm{h}(\mathrm{L}) & \cdots & \mathrm{h}(1) \\ 0 & 0 & \cdots & 0 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \mathrm{h}(\mathrm{L}) \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & 0 \end{bmatrix}_{S\times S} \\ &= \mathbf{H}_{\mathrm{ISI}} + z^{-1}\mathbf{H}_{\mathrm{IBI}}, \end{aligned} \tag{5.6}$$

in which $\mathbf{H}_{\mathrm{ISI}}$ and $\mathbf{H}_{\mathrm{IBI}}$ represent ISI and IBI effects produced by the wireless channel [15]. This model is valid when the transmitted OFDM symbol's length exceeds the channel order, that is when $S > L$.

By using equation (5.6), the received signal in the time domain is

$$\mathbf{y}(k) = \mathbf{H}_{\mathrm{ISI}}\mathbf{u}(k) + \mathbf{H}_{\mathrm{IBI}}\mathbf{u}(k-1) + \mathbf{v}(k), \tag{5.7}$$

where $\mathbf{v}(k)$ is an additive Gaussian noise vector with zero mean and covariance matrix $\sigma_v^2\mathbf{I}_S$. The receiver removes the redundancy by multiplying $\mathbf{y}(k)$ by $\mathbf{R} \in \mathbb{C}^{N\times S}$, where

$$\mathbf{R} = \begin{cases} \mathbf{R}_{\mathrm{ZP}} = \begin{bmatrix} \mathbf{I}_N & \begin{matrix} \mathbf{I}_K \\ \mathbf{0}_{(N-K)\times K} \end{matrix} \end{bmatrix}, \text{ for zero-padding,} \\ \mathbf{R}_{\mathrm{CP}} = \begin{bmatrix} \mathbf{0}_{N\times K} & \mathbf{I}_N \end{bmatrix}, \text{ for cyclic-prefix.} \end{cases} \tag{5.8}$$

In the absence of noise, equation (5.7) becomes

$$\mathbf{y}(k) = \mathbf{R}\mathbf{H}_{\mathrm{ISI}}\mathbf{A}\mathbf{x}(k) + \mathbf{R}\mathbf{H}_{\mathrm{IBI}}\mathbf{A}\mathbf{x}(k-1). \tag{5.9}$$

If sufficient redundancy is inserted $K \geq L$, then

$$\mathbf{R}\mathbf{H}_{\mathrm{IBI}}\mathbf{A} = \mathbf{H}_{\mathrm{r}} = \mathbf{0}_N, \tag{5.10}$$

which means that IBI is eliminated for both ZP-OFDM-OLA (zero padding-OFDM-overlap and add) and CP-OFDM (cyclic prefix-OFDM). Moreover,

$$\mathbf{H}_c = \mathbf{R}\mathbf{H}_{\mathrm{ISI}}\mathbf{A} \tag{5.11}$$

is a circulant matrix. Hence, equation (6.4) becomes

$$\mathbf{y}(k) = \mathbf{H}_c\mathbf{x}(k) = \mathbf{h}_N \circledast \mathbf{x}(k), \tag{5.12}$$

in which $\mathbf{h}_N$ is the first column of $\mathbf{H}_c$ and $\circledast$ is the circular convolution operator. By using the circular convolution theorem, we can employ the FFT to transform the circular convolution into component-wise multiplication. Hence, the least squares (LS) channel estimate $\widehat{\boldsymbol{h}}_{\mathrm{LS}} \in \mathbb{C}^{N_P \times 1}$ can be written for each subcarrier x

$$\widehat{h}_{\mathrm{LS}}(n) = \frac{y_P(n)}{x_P(n)} \quad \text{for} \quad n \in \mathbb{B}_P, \tag{5.13}$$

where $\boldsymbol{x}_P \in \mathbb{C}^{N_P \times 1}$ and $\boldsymbol{y}_P \in \mathbb{C}^{N_P \times 1}$ are the transmitted and received pilot signals in the frequency domain.

To obtain channel state information (CSI), we can also employ the linear minimum mean-squared error (LMMSE) estimator

$$\widehat{\boldsymbol{h}}_{\mathrm{LMMSE}} = \mathbf{W}_{\mathrm{LMMSE}}\widehat{\boldsymbol{h}}_{\mathrm{LS}}, \tag{5.14}$$

where $\mathbf{W}_{\mathrm{LMMSE}}$ is the LMMSE weight matrix defines as follows

$$\mathbf{W}_{\mathrm{LMMSE}} = \mathbf{R}_{\boldsymbol{h}_N\widehat{\boldsymbol{h}}_{\mathrm{LS}}} \left(\mathbf{R}_{\boldsymbol{h}_N\boldsymbol{h}_N} + \frac{\sigma_v^2}{E[||\mathbf{x}||^2]}\mathbf{I}_N \right)^{-1}, \tag{5.15}$$

in which $\mathbf{R}_{\boldsymbol{h}_N\widehat{\boldsymbol{h}}_{\mathrm{LS}}}$ is the cross-correlation matrix between the true channel vector and channel estimate vector in the frequency domain. The auto-correlation matrix of $\boldsymbol{h}_N$ is $\mathbf{R}_{\boldsymbol{h}_N\boldsymbol{h}_N}$. The energy of the transmitted symbol is $E[||\mathbf{x}||^2]$, and $\boldsymbol{h}_N$ is $\mathbf{h}_N$ in the frequency domain. This solution has some practical forms of implementation, see [143].

Since $N_P = N_D = N$ for block-type pilot arrangement, $\widehat{\mathbf{h}} \in \mathbb{C}^{N \times 1}$. For comb-type pilot arrangement, $N = N_P + N_D$, and hence we need to interpolate the channel estimate $\widehat{\mathbf{h}} \in \mathbb{C}^{N_P \times 1}$ to obtain the whole channel estimation $\widehat{\mathbf{h}} \in \mathbb{C}^{N \times 1}$.

5.2.2 Equalization

After channel estimation, $\widehat{\mathbf{h}} \in \mathbb{C}^{N\times 1}$ is used to perform equalization and recover the transmitted data signals, as also shown in Figure 5.1.

For ZP-OFDM-OLA and CP-OFDM, the received data signal $\mathbf{y}(k)$ is multiplied by the receiver matrix in equation (5.8). After performing the FFT, the estimated signal is

$$\boldsymbol{y}'(k) = \mathbf{E}\mathbf{W}_N\mathbf{H}_c\mathbf{W}_N^{\mathrm{H}}\mathbf{x}(k) + \mathbf{E}\mathbf{W}_N\mathbf{R}\mathbf{v}(k), \tag{5.16}$$

in which $\mathbf{E} = (\mathbf{W}_N\mathbf{H}_c\mathbf{W}_N^{\mathrm{H}})^{-1}$ is the basic frequency domain equalization (BFDE) matrix. If $K \geq L$, $\mathbf{H}_c$ is a circulant matrix, and it can be diagonalized by using the DFT and IDFT matrices. Therefore, $\mathbf{E} = \boldsymbol{\Lambda}^{-1}$, where $\boldsymbol{\Lambda}$ is a diagonal matrix with the FFT of $\mathbf{h}_N = [\mathbf{h}\ \ \mathbf{0}_{(N-L-1)\times 1}]^{\mathrm{T}}$ as diagonal [15]. Hence, equation (5.16) can be written for each subcarrier as

$$y'(n) = \frac{y(n)}{\widehat{h}_{\mathrm{LS}}(n)} \quad \text{for} \quad n = 1, \cdots N, \tag{5.17}$$

in which $\boldsymbol{y} = \mathbf{W}_N\mathbf{H}_c\mathbf{W}_N^{\mathrm{H}}\mathbf{x} + \mathbf{W}_N\mathbf{R}\mathbf{v}(k)$. After equalization, the resulting symbols $\boldsymbol{y}'$ can be demodulated and the received bits estimated.

For ZPZJ-OFDM, the receiver matrix is defined as

$$\mathbf{R}_{\mathrm{ZPZJ}} = \left[\, \mathbf{0}_{N\times(L-K)}\ \overline{\mathbf{G}} \,\right] \in \mathbb{C}^{N\times S}, \tag{5.18}$$

where $\overline{\mathbf{G}} \in \mathbb{C}^{N\times(N+2K-L)}$ is the equalization matrix. As in the ZP-OFDM-OLA, if $K \geq L$, $\mathbf{R}_{\mathrm{ZPZJ}}\mathbf{H}_{\mathrm{IBI}}\mathbf{A}_{\mathrm{ZP}} = \mathbf{0}_N$, and the IBI is eliminated. However, the matrix product

$$\mathbf{R}_{\mathrm{ZPZJ}}\mathbf{H}_{\mathrm{ISI}}\mathbf{A}_{\mathrm{ZP}} = \overline{\mathbf{H}} \in \mathbb{C}^{(N+2K-L)\times N} \tag{5.19}$$

results in a rectangular Toeplitz matrix which is not easily diagonalized as in the ZP-OFDM-OLA and CP-OFDM cases. For the ZPZJ-OFDM, the equalizer can be defined as in [15],

$$\overline{\mathbf{G}} = \begin{cases} (\overline{\mathbf{H}}^{\mathrm{H}}\ \overline{\mathbf{H}})^{-1}\overline{\mathbf{H}}^{\mathrm{H}}, \text{ for ZF equalization,} \\ (\overline{\mathbf{H}}^{\mathrm{H}}\ \overline{\mathbf{H}} + \frac{\sigma_v^2}{\sigma_x^2}\mathbf{I}_N)^{-1}\overline{\mathbf{H}}^{\mathrm{H}}, \text{ for MMSE equalization.} \end{cases} \tag{5.20}$$

Another possible equalizer is

$$\overline{\mathbf{G}} = (\overline{\mathbf{H}}^{\mathrm{H}}\ \overline{\mathbf{H}} + \lambda\mathbf{I}_N)^{-1}\overline{\mathbf{H}}^{\mathrm{H}}, \tag{5.21}$$

in which λ is a constant regularization factor. After equalization, we perform the FFT. Then, the resulting symbols $\boldsymbol{y}'$ can be demodulated, and the received bits estimated.

5.3 Basic simulation setup

When not mentioned, the following parameters and models are used.

- The OFDM system has $N = 64$ subcarriers;
- The input symbols are 16-QAM samples;
- The block-type and comb-type pilot arrangements are considered in the experiments of Subsections 5.4.2.3 and 5.4.3.1, respectively;
- The ITU Pedestrian A channel [144] is generated with Matlab's `stdchan` using the channel model `itur3GPAx` with a carrier frequency $f_c = 2$ GHz, 4 km/h as velocity, $T_s = 200$ ns and order $L = 10$;
- The NNs are trained for each signal-to-noise ratio (SNR);
- The bit error rate (BER) is presented as a function of the SNR by averaging 2000 independent runs;
- The MSE is also presented as a function of the SNR by averaging 2000 independent runs.

5.4 OFDM systems with insufficient redundancy using NNs

The use of redundancy in OFDM with $K \geq L$ not only prevents the IBI but also transforms the linear convolutions into circular convolutions, leading to multiplication in the frequency domain [145]. If the OFDM system operates with insufficient redundancy $K < L$, $\mathbf{H}_r \neq \mathbf{0}_N$ in equation (5.10) and $\mathbf{H}_c$ in equation (5.11) is no longer circulant. The ZF solution is then impossible for time-invariant transceivers, with the exception of ZPZJ. Our goal here is to deliver ML-based solutions to cope with the case in which the transmitter adds insufficient redundancy to the OFDM symbol.

In this section, we start by using the channel estimator (CE) subnet proposed in [125, 130] to obtain a refined channel estimate. Then, we propose an improved channel estimator (ICE) subnet in an attempt to cope with nonlinear distortions, and the insufficient redundancy. We also propose a subnet to improve the symbol detection, which we call a symbol detector minimum redundancy (SDMR) subnet.

5.4.1 'CE + SDMR' solution for block-type pilot arrangement in CP-OFDM systems

We started by defining the minimum redundancy (MR) OFDM receiver allowing the ZF solution to have a total redundancy length equal to $L/2$. The proposed MR OFDM receiver is composed of two subnets. The first subnet is the same used in [125, 130]. The CE subnet is responsible for obtaining a refined channel estimate. The second one is the SDMR subnet, that aims to detect the received symbols. The solution 'CE + SDMR' is proposed in our previous work for CP-OFDM [21].

5.4.1.1 CE subnet

The CE subnet is first proposed in [130]. Inspired by equation (5.14), the CE subnet utilizes a two-layer neural network to obtain a refined channel estimation [130], as depicted in Figure 5.3. The LS estimate described in equation (5.13) is converted into a real-valued block

$$\widetilde{\boldsymbol{h}}_{\text{LS}} = \begin{bmatrix} \text{Re}\{\widehat{\boldsymbol{h}}_{\text{LS}}\} \\ \text{Im}\{\widehat{\boldsymbol{h}}_{\text{LS}}\} \end{bmatrix} \tag{5.22}$$

and used as input of the CE subnet. The real-valued block version of a vector $\mathbf{a}$ is defined as

$$\widetilde{\mathbf{a}} = \begin{bmatrix} \text{Re}\{\mathbf{a}\} \\ \text{Im}\{\mathbf{a}\} \end{bmatrix}. \tag{5.23}$$

To accelerate the convergence speed, the CE subnet is initialized by the real-valued LMMSE weight matrix

$$\widetilde{\mathbf{W}}_{\text{LMMSE}} = \begin{bmatrix} \text{Re}\{\mathbf{W}_{\text{LMMSE}}\} & -\text{Im}\{\mathbf{W}_{\text{LMMSE}}\} \\ \text{Im}\{\mathbf{W}_{\text{LMMSE}}\} & \text{Re}\{\mathbf{W}_{\text{LMMSE}}\} \end{bmatrix}, \tag{5.24}$$

in which $\mathbf{W}_{\text{LMMSE}}$ is defined in equation (5.15). As in [130], we follow the method in [142] to compute the LMMSE matrix $\mathbf{W}_{\text{LMMSE}}$ defined in equation (5.14).

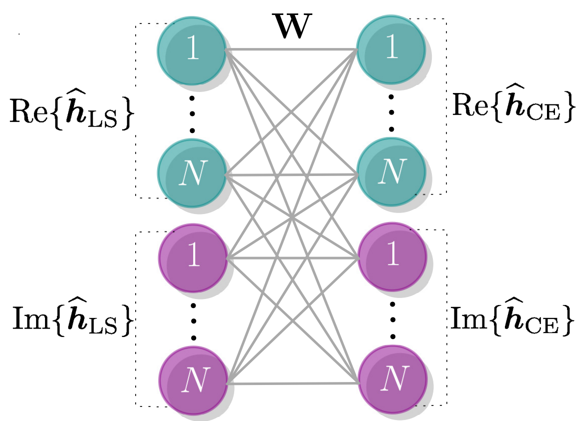


Figure 5.3: CE subnet.

The CE subnet is trained by minimizing the mean squared error (MSE) between a noisy version of the actual channel in the frequency domain $\overline{\boldsymbol{h}}_N$ and the prediction $\widehat{\boldsymbol{h}}_{\text{CE}}$ by using the adaptive moment estimator (Adam) optimizer. Since obtaining the true channel response is difficult in practice, these training labels are more suitable than the ones used in [125, 130]. We define $\overline{\boldsymbol{h}}_N = \boldsymbol{h}_N + \boldsymbol{v}$, where $\boldsymbol{h}_N$ is the FFT of the true channel impulse response, and $\boldsymbol{v}$ is an additive Gaussian noise vector with zero mean and covariance matrix $\sigma_v^2 \mathbf{I}_N$. The learning rate is set to $\mu = 0.001$. The training and testing sets contain 3000 and 1000 samples, respectively. The batch size and epochs are set to 50 and 200, respectively.

5.4.1.2 SDMR subnet

The proposed symbol detection subnetwork is a neural network with three hidden layers, as illustrated in Figure 5.4.

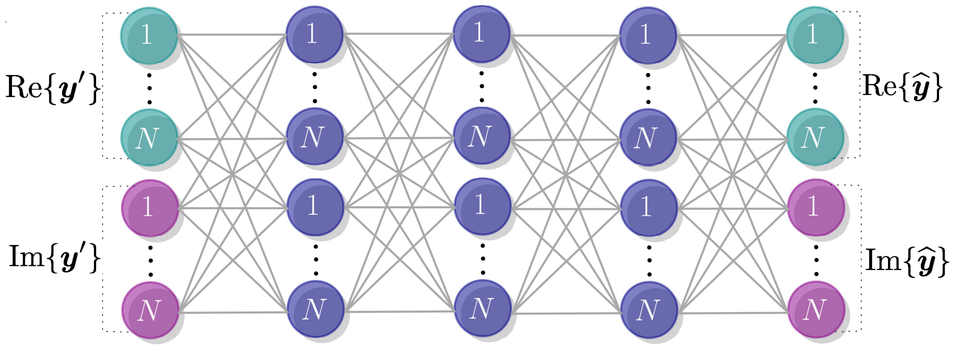

Re{$\boldsymbol{y}'$}
Im{$\boldsymbol{y}'$}
Re{$\widehat{\boldsymbol{y}}$}
Im{$\widehat{\boldsymbol{y}}$}
1
N

Figure 5.4: Proposed SDMR subnet.

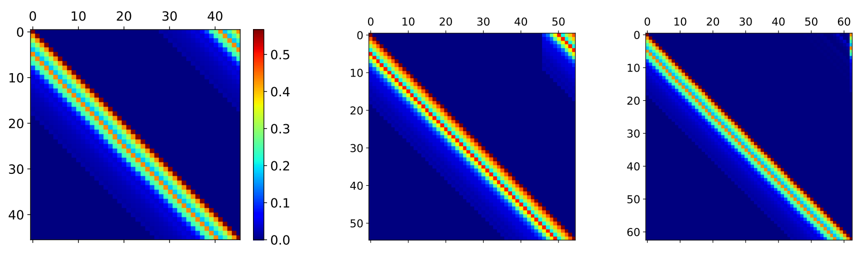
The hidden layers have as activation function the hyperbolic tangent. In addition, the SDMR subnet utilizes a linear activation function at the output layer. Unlike [129, 130] works concerning bits estimation in a classification problem formulation, the SDMR subnet output layer has been chosen as a linear activation function since it provides a better estimation for the symbols as a regression problem.

We first obtain the frequency domain equalizer solution,

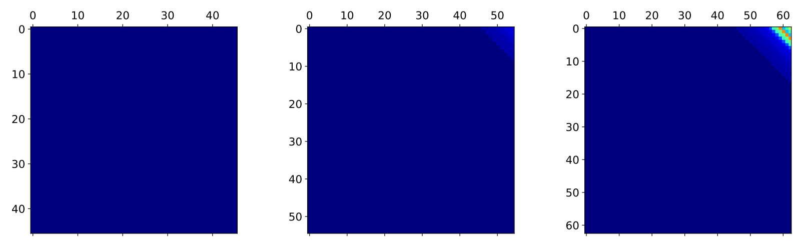
$$y'_D(n) = \frac{y_D(n)}{\widehat{h}_{\text{CE}}(n)} \quad \text{for} \quad n = 1, \cdots N, \tag{5.25}$$

in which $\widehat{\boldsymbol{h}}_{\text{CE}}$ is the channel estimate obtained in the CE subnet. Then, the real-valued block version of $\boldsymbol{y}'_D$ is used as input in the SDMR subnet. The real-valued block version of a vector is defined in equation (5.23). The output $\widehat{\boldsymbol{x}}$ is then converted to a complex vector and demodulated to obtain the estimated bits.

A basic FDE or BFDE provides a simple technique to mitigate the distortion due to a frequency selective channel. For the linear channel model, the use of a cyclic prefix in OFDM with $K \geq L$ not only prevents the IBI but also transforms the linear

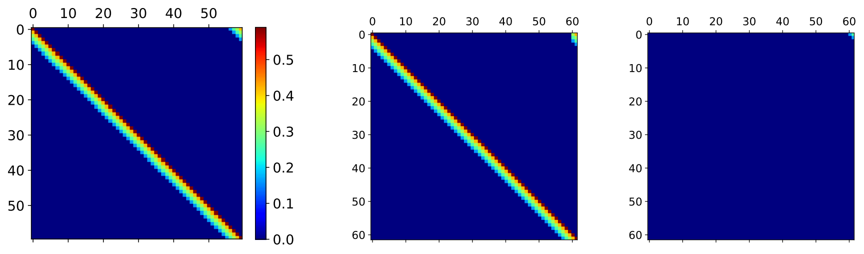


(a) $\mathbf{H}_c$, $K = L$ (b) $\mathbf{H}_c$, $K = L/2$ (c) $\mathbf{H}_c$, $K = 1$



(d) $\mathbf{H}_r$, $K = L$ (e) $\mathbf{H}_r$, $K = L/2$ (f) $\mathbf{H}_r$, $K = 1$

Figure 5.5: Illustration of matrices $\mathbf{H}_c$ and $\mathbf{H}_r$ for $S = 64, L = 18$ when K is varied.



(a) $\mathbf{H}_c$, $K = L$ (b) $\mathbf{H}_c$, $K = L/2$ (c) $\mathbf{H}_r$, $K = L/2$

Figure 5.6: Illustration of matrices $\mathbf{H}_c$ and $\mathbf{H}_r$ for $S = 64, L = 4$ when K is varied.

convolutions into circular convolutions, which are equivalent to multiplication in the frequency domain [145]. Consider now using CP with $K = L/2$. In this case, $\mathbf{H}_c$ in equation (5.11) is no longer circulant, $\mathbf{H}_r \neq \mathbf{0}_N$, and hence the BFDE does not provide the ZF solution, with the exception of ZPZJ. Figures 5.5 and 5.6 illustrate the impact on matrices $\mathbf{H}_c$ and $\mathbf{H}_r$ when K is reduced. However, the performance is improved as we employ a DNN after the BFDE to detect the symbols. We can then use a reduced CP length and save some of the subcarriers to transmit information data.

The SDMR subnet is also trained by minimizing the mean squared error (MSE) between the transmitted symbol $\boldsymbol{x}$ and the prediction $\widehat{\boldsymbol{x}}$ by using the Adam optimizer. The learning rate is set to $\mu = 0.0001$. The training and testing sets contain 3000 and 1000 samples, respectively. The batch size and epochs are set to 50 and 600, respectively.

5.4.1.3 Simulation results considering the 'CE+SDMR' solution

In this section, we consider the basic simulation setup in Section 5.3, with some exceptions mentioned as follows. The input symbols are 16-QAM or 64-QAM samples. As in [125, 129, 130], the channel is modeled by the wireless world initiative for new radio (WINNER II) [146] under urban scenarios. We also consider a scenario in which a nonlinearity, modeled by the hyperbolic tangent function, is present at the transmitter output. Furthermore, we analyze the case where the channel order is small compared to the transmit block size. All the simulation results are obtained by averaging $T = 500$ independent runs.

The considered methods are presented in more detail in Table 5.1, as well as the main OFDM system. Unfortunately, the methods proposed in [125, 130] were not evaluated in the simulations since the information needed to perform a fair comparison is not available. However, it should be mentioned that we explore our proposal in more encompassing situations. The nonlinearity is not known to the 'Exact_wCP' method when it is present in the transmitter. In this case, the 'Exact_wCP' method ignores the nonlinearity and considers the exact linear-channel model.

Figure 5.7 depicts the average BER for a CP-OFDM receiver with a 16-QAM under the linear scenario. The performance of the proposed MR-net with reduced CP is closer to the lower bound, Exact_CP. Moreover, MR-net outperforms the conventional OFDM receiver (LMMSE_wCP). By comparing the proposed method with reduced CP and its CP-free version, we can observe a considerable gain in performance. This unveils the possibility of applying the minimum redundancy technique with other DNN-based receivers in OFDM.

The proposed method performs reasonably well under higher modulations schemes like 64-QAM, as illustrated in Figure 5.8.

Table 5.1: Methods evaluated in the simulations considering 'CE+SDMR' configuration and CP-OFDM systems.

Method	**CP length**	**Channel estimation**	**Equalization**
MR-net_wCP_red	$K = L/2$	CE subnet	ZF+SDMR
LS_wCP_red	$K = L/2$	LS	ZF
LMMSE_wCP	$K = L$	LMMSE	ZF
Exact_wCP	$K = L$	Exact	ZF
CE + ZF_nCP	$K = 0$	CE subnet	ZF
MR-net_nCP	$K = 0$	CE subnet	ZF+SDMR
LS_nCP	$K = 0$	LS	ZF

Figure 5.9 depicts the average BER for a CP-OFDM system operating with a 16-QAM modulation scheme when simple nonlinearity (hyperbolic tangent) is present. In this case, the conventional method LMMSE_wCP achieves worse BER levels as the nonlinearity is unknown. On the other hand, the performance achieved by the proposed method MR-net remains closer to the lower bound.

Figure 5.10 depicts the BER results when the channel order is reduced to $L = 4$ and, as in the scenario considered in Figure 5.8, a nonlinearity is present in the transmitter. In this case, the proposed methods outperform the competing methods, whereas the Exact_wCP shows resilience to the nonlinearity effect applied to the transmitted symbols.

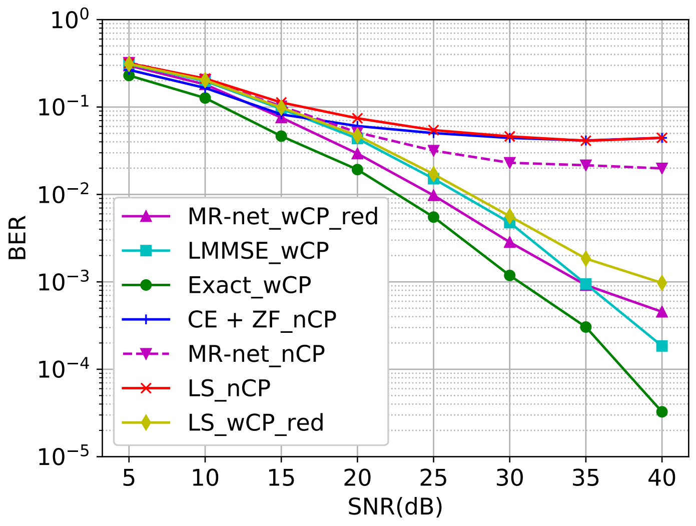


Figure 5.7: Average BER for a minimum redundancy CP-OFDM receiver with 64 subcarriers. 16-QAM symbols are transmitted through a channel modeled using WINNER II under urban scenarios in which the channel order is $L = 18$. The scenario is linear, that is, no nonlinear distortion is presented in the transmitter. $K = 18$ for methods with 'wCP', $K = 9$ for methods with 'wCP_red', and $K = 0$ for methods with 'nCP'.

5.4.2 'ICE + SDMR' solution for the block-type pilot arrangement in CP and ZP-OFDM systems

In this section, we extended the work in [21] by proposing the improved channel estimator (ICE) subnet in [22]. The ICE aims to undo the nonlinear clipping distortion while dealing with insufficient redundancy.

5.4.2.1 ICE subnet

The ICE subnet is inspired by the CE subnet, which utilizes a two-layer neural network to obtain a refined channel estimation [130]. As illustrated in Figure 5.11, the ICE subnet has one hidden layer with a hyperbolic tangent as the activation function.

The real-valued block version of the complex LS channel estimate $\widehat{\boldsymbol{h}}_{LS} \in \mathbb{C}^{N\times 1}$ is the input of the ICE subnet. The real-valued block version of a vector is defined

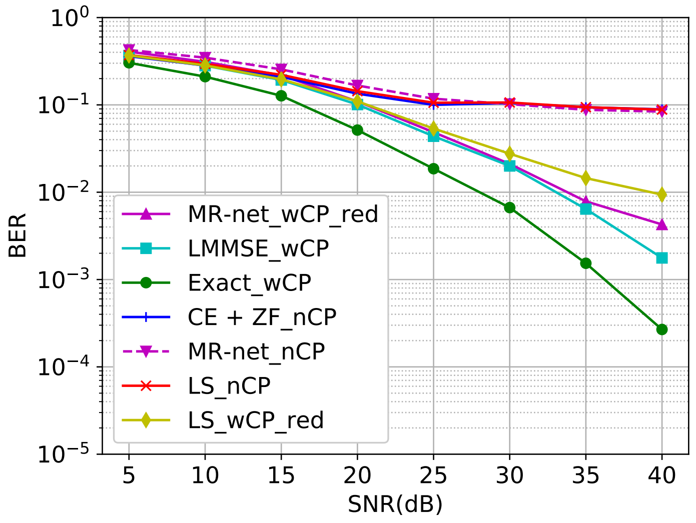


Figure 5.8: Average BER for a minimum redundancy CP-OFDM receiver with 64 subcarriers. 64-QAM symbols are transmitted through a channel modeled using WINNER II under urban scenarios in which the channel order is $L = 18$. The scenario is linear, that is, no nonlinear distortion is present in the transmitter. $K = 18$ for methods with 'wCP', $K = 9$ for methods with 'wCP_red', and $K = 0$ for methods with 'nCP'.

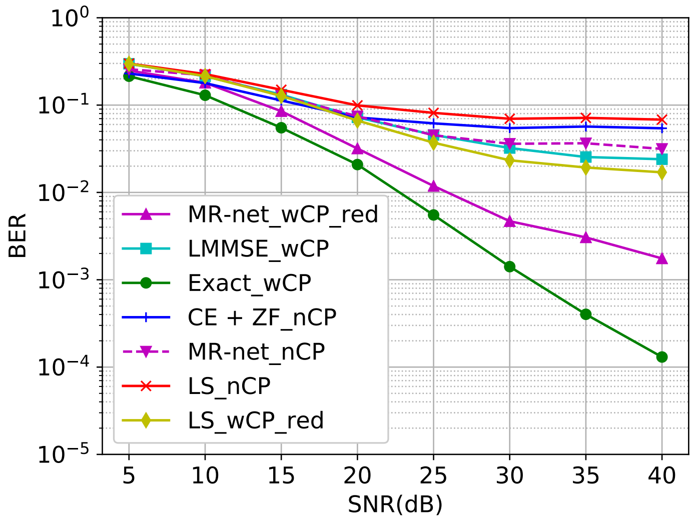


Figure 5.9: Average BER for a minimum redundancy CP-OFDM receiver with 64 subcarriers. 16-QAM symbols are transmitted through a channel modeled using WINNER II under urban scenarios in which the channel order is $L = 18$. The scenario is nonlinear, that is, a nonlinear distortion (hyperbolic tangent) is present in the transmitter. $K = 18$ for methods with 'wCP', $K = 9$ for methods with 'wCP_red', and $K = 0$ for methods with 'nCP'.

in equation (5.23).

The ICE subnet is trained by minimizing the mean squared error (MSE) between a noisy version of the actual channel in the frequency domain $\overline{\boldsymbol{h}}_N$ and the prediction

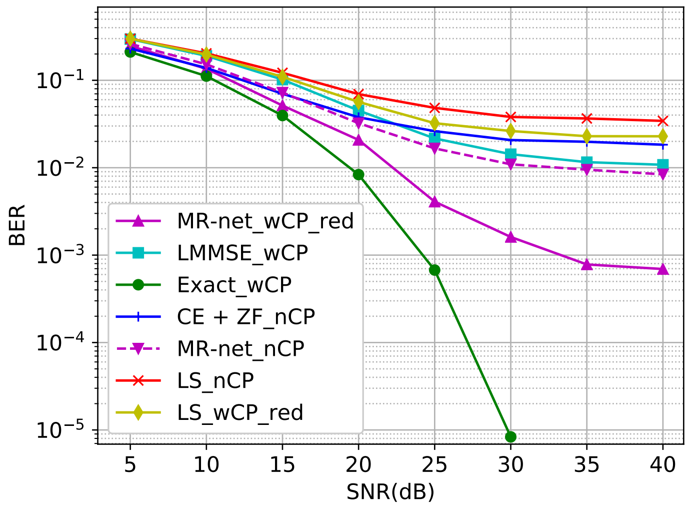
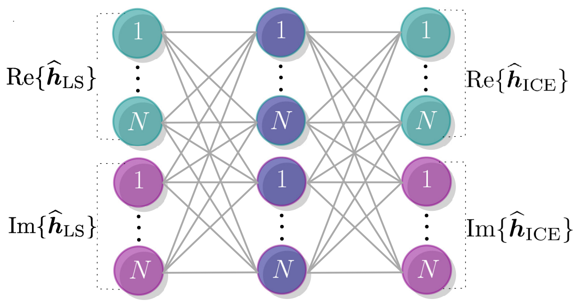


Figure 5.10: Average BER for a minimum redundancy CP-OFDM receiver with 64 subcarriers. 16-QAM symbols are transmitted through a channel modeled using WINNER II under urban scenarios in which the channel order is reduced to $L = 4$.The scenario is nonlinear, that is, a nonlinear distortion (hyperbolic tangent) is present in the transmitter. $K = 4$ for methods with 'wCP', $K = 2$ for methods with 'wCP_red', and $K = 0$ for methods with 'nCP'.



$\mathrm{Re}\{\widehat{\boldsymbol{h}}_{\mathrm{LS}}\}$
$\mathrm{Im}\{\widehat{\boldsymbol{h}}_{\mathrm{LS}}\}$
$\mathrm{Re}\{\widehat{\boldsymbol{h}}_{\mathrm{ICE}}\}$
$\mathrm{Im}\{\widehat{\boldsymbol{h}}_{\mathrm{ICE}}\}$
1
N

Figure 5.11: Proposed ICE subnet.

$\widehat{\boldsymbol{h}}_{\mathrm{CE}}$ by using the adaptive moment estimator (Adam) optimizer with learning rate $\mu = 0.001$. Since obtaining the true channel response is difficult in practice, the noisy channel training labels are more suitable than the true channel response labels used in [125, 130]. The true channel response labels consist of the real-valued block version of $\boldsymbol{h}_N$. We then define the labels as the real-valued block version of

$$\overline{\boldsymbol{h}}_N = \boldsymbol{h}_N + \boldsymbol{v}, \tag{5.26}$$

where $\boldsymbol{v}$ is an additive Gaussian noise vector with zero mean and covariance matrix $\sigma_v^2\mathbf{I}_N$. The real-valued block version of a vector is defined in equation (5.23). The training set contains 3000 samples. The mini-batch size comprises 50 samples, and 200 epochs are required to train the ICE subnet. At each epoch, 60 iterations are required to explore the entire dataset. As we generate the training data set, we produce enough data samples to learn the problem. Moreover, to deliver a certain performance when using the model, we run Monte Carlo simulations.

With the channel estimate produced by the ICE subnet $\widehat{\boldsymbol{h}}_{\text{ICE}}$, we can obtain the frequency domain equalizer solution,

$$y'_D(n) = \frac{y_D(n)}{\widehat{h}_{\text{ICE}}(n)} \quad \text{for} \quad n \in \mathbb{B}_D, \tag{5.27}$$

for the data subcarriers.

5.4.2.2 SDMR subnet

As we are dealing with insufficient redundancy, we propose the second block composed of the SDMR subnet, introduced in Subsection 5.4. The only difference is that the SDMR is trained considering the channel estimate obtained with the ICE subnet. Therefore, we suggest employing the second block in Figure 5.12 when the prefix length is too short and severely degrades the BER performance.

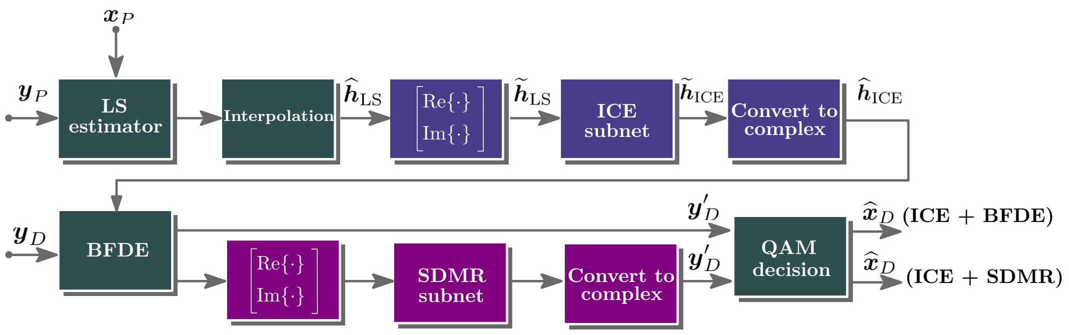

$\boldsymbol{x}_P$
$\boldsymbol{y}_P$
LS estimator
Interpolation
$\widehat{\boldsymbol{h}}_{\text{LS}}$
Re{·} Im{·}
$\widetilde{\boldsymbol{h}}_{\text{LS}}$
ICE subnet
$\widetilde{\boldsymbol{h}}_{\text{ICE}}$
Convert to complex
$\widehat{\boldsymbol{h}}_{\text{ICE}}$
$\boldsymbol{y}_D$
BFDE
$\boldsymbol{y}'_D$
Re{·} Im{·}
SDMR subnet
Convert to complex
$\boldsymbol{y}'_D$
QAM decision
$\widehat{\boldsymbol{x}}_D$ (ICE + BFDE)
$\widehat{\boldsymbol{x}}_D$ (ICE + SDMR)

Figure 5.12: Proposed OFDM receivers.

After the BFDE, the real-valued block version of $\boldsymbol{y}'_D$ is used as input for the SDMR subnet illustrated in Figure 5.4. The real-valued block version of a vector is defined in equation (5.23). The output $\widehat{\boldsymbol{y}}_D$ is then converted to a complex vector, processed using the QAM demodulator to obtain the estimated symbols $\widehat{\boldsymbol{x}}_D$.

As shown in [147, 148], a ZF solution is achievable in ZPZJ-OFDM if $\overline{\mathbf{H}}$ in equation (5.19) is full-rank, then inequality

$$N \leq \min\{(N + 2K - L),N\} \tag{5.28}$$

must be satisfied, which implies that

$$N \leq N + 2K - L \rightarrow K \geq L/2. \tag{5.29}$$

Although efficient processing of the inverse of $\overline{\mathbf{H}}$ is demonstrated in [126, 127], if $\overline{\mathbf{H}}$ is ill-conditioned, its inverse is still difficult to compute. Consider now using $K = L/2$

in ZP-OFDM-OLA. In this case, $\mathbf{H}_c$ is no longer circulant, $\mathbf{H}_r \neq \mathbf{0}_N$, and hence the BFDE does not provide the ZF solution. However, as we employ a NN after the BFDE to detect the symbols, the performance is improved. This allows the use of a reduced ZP length, alleviating the necessity of knowing the channel order exactly. Furthermore, some of the subcarriers are saved for transmitting information data.

The learning curves for the SDMR subnet are shown in Figure 5.13 for $K = 0$ and $K = L/2 = 5$, for example. The loss is defined as the MSE between the transmitted symbol $\boldsymbol{x}$ and the prediction $\widehat{\boldsymbol{x}}$. Observe that the generalization gap between train and validation curves is more significant when $K = 5$. It might indicate that the training dataset does not provide the necessary information to learn the problem when sufficient redundancy is added at the transmitter. That is when redundancy length is long enough; the SDMR subnet is unnecessary.

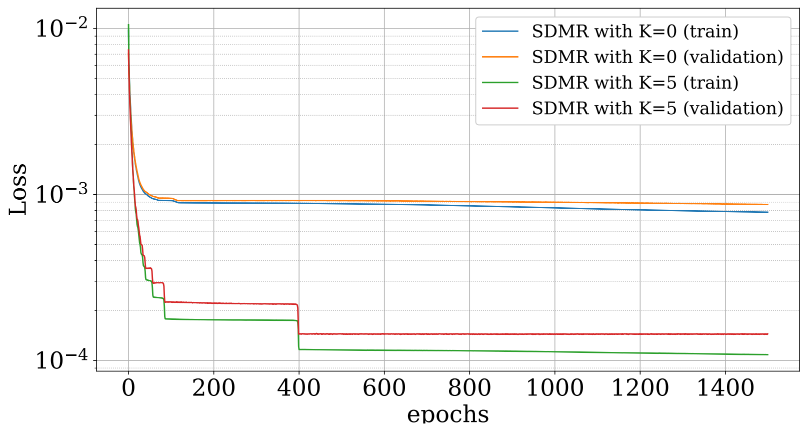


Figure 5.13: Learning curves for SDMR subnet.

5.4.2.3 Simulation results considering the 'ICE+SDMR' solution

In this section, we consider the basic simulation setup in Section 5.3. The tested methods consist of one channel estimator and one equalizer. For example, 'LS+BFDE' represents the LS channel estimator in equation (5.13) and the BFDE in equation (5.17). 'Exact' means that the receiver has perfect CSI. 'LMMSE' represents the LMMSE channel estimator in equation (5.14), where Perfect CSI is assumed to compute the correlation matrices. We consider the LMMSE solution as a standard comparison method. 'ICE' is our proposed channel estimator, and 'SDMR' consists of our proposed equalizer/symbol detector. Moreover, 'ZP' stands for ZP-OFDM-OLA receiver, whereas 'ZPZJ' denotes ZPZJ-OFDM receiver. Regarding the ZPZJ-OFDM, the equalizer 'MMSE' is used when $K = L$, and 'reg' is used when $K = L/2$. 'MMSE is defined in equation (5.20), and 'reg' in equation (5.21) with $\lambda = 10^{-3}$.

In the nonlinear scenario, we include the nonlinear clipping distortion with clipping ratio $C_R = 1.3$ at the transmitter side. The nonlinear clipping distortion is not known to the 'Exact' method. Therefore, the 'Exact' method ignores the nonlinearity and considers the exact linear-channel model. Besides, the linear scenario is defined by using $C_R \to \infty$.

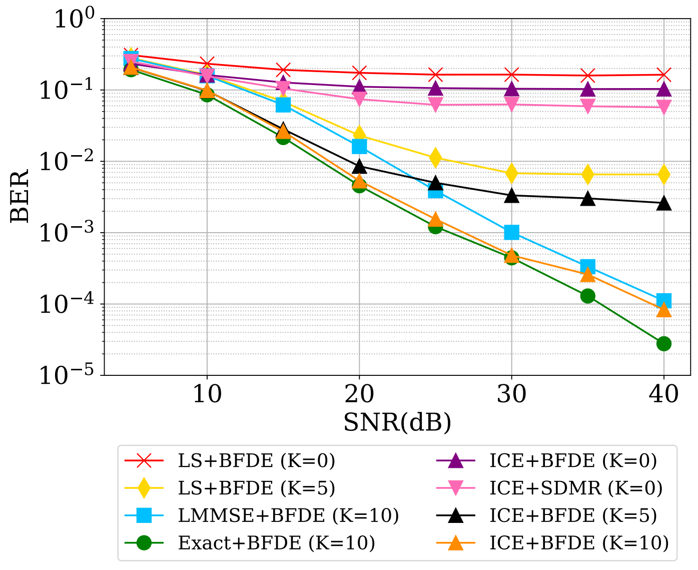
Figure 5.14a depicts the average BER for CP-OFDM receivers with a block-type pilot arrangement under the linear scenario, that is, when $C_R \to \infty$. When the CP length is equal to the channel order $L = 10$, the proposed 'ICE+BFDE' outperforms the conventional OFDM receiver 'LMMSE'. Moreover, the 'ICE+BFDE' performance remains quite close to the lower bound 'Exact + BFDE'. By reducing the CP length to $L/2 = 5$, the proposed 'ICE+BFDE' outperforms the 'LMMSE' receiver for SNR < 25 dB with the benefit of requiring fewer redundancy elements. When redundancy is not employed, the proposed 'ICE+BFDE' performance is improved by the SDMR subnet, resulting in the proposed 'ICE+SDMR'.

Since the nonlinear clipping distortion is present at the transmitter side, the 'Exact' channel estimator does not know this nonlinearity. As shown in Figure 5.14b, the proposed ICE subnet handles the nonlinear distortion. It even outperforms the 'Exact' channel estimator with only $K = 5$ redundant elements when $K = L = 10$ for the considered CP-OFDM receivers with the block-type pilot arrangement.

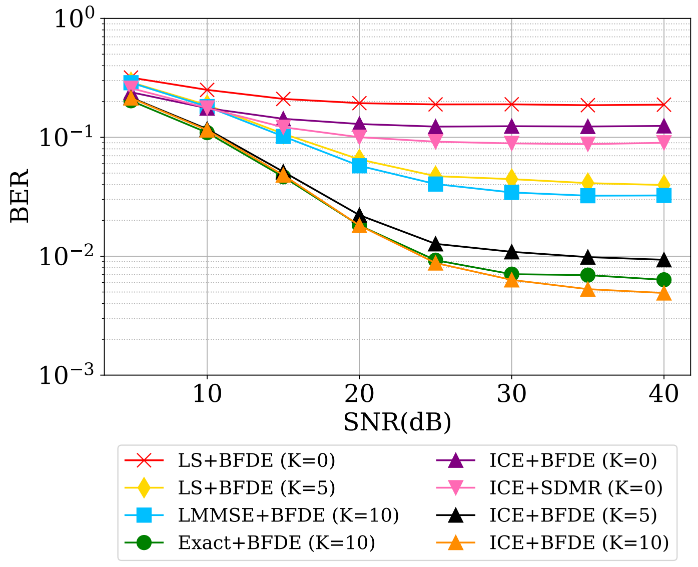
Figure 5.15 compares the MSE between the true channel and the estimated channel using ICE, CE, and LS channel estimators for CP-OFDM receivers with the block-type pilot arrangement. The ICE subnet outperforms the CE subnet, especially for the nonlinear scenario in Figure 5.15b. In terms of BER, as shown in Figure 5.16, the ICE subnet performs quite close to the CE subnet when $C_R \to \infty$, whereas the ICE subnet significantly outperforms the CE subnet for $C_R = 1.3$.

Figure 5.17 depicts the BER curves for the linear scenario, that is, when the transmitter is free from nonlinear distortions, for the considered ZP-OFDM receivers with the block-type pilot arrangement. We consider the channel model described in the basic setup in Section 5.3 and the WINNER II channel model tested in Subsection 5.4.1.

For both ITU Pedestrian A and WINNER II, the combination of 'ICE+SDMR' provides an improvement only for ZP-OFDM-OLA when $K = 0$. When $K = L/2$, the ICE subnet is adequate to make both ZPZJ-OFDM and ZP-OFDM-OLA outperform the LMMSE channel estimator for SNR < 25dB with the benefit of requiring less redundant elements, as shown in Figure 5.17a. As depicted in Figure 5.17b for the WINNER II channel, $\overline{\mathbf{H}}$ can be ill-conditioned when ZPZJ is operating with $K = L/2$, and the BER performance degrades. By using the ICE subnet with $K = L$, ZPZJ-OFDM and ZP-OFDM-OLA approximate or even outperform the ZP-OFDM-OLA with perfect CSI.



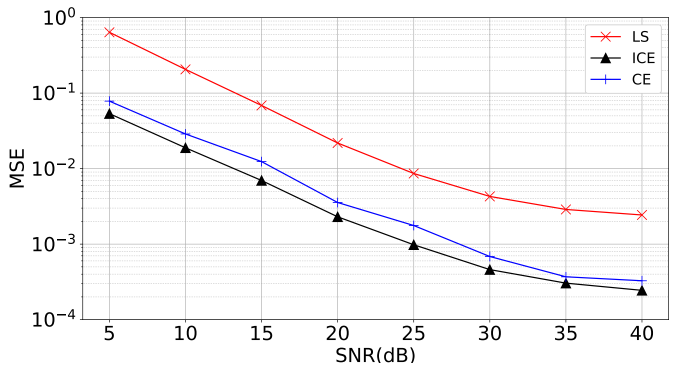
(a) Linear scenario



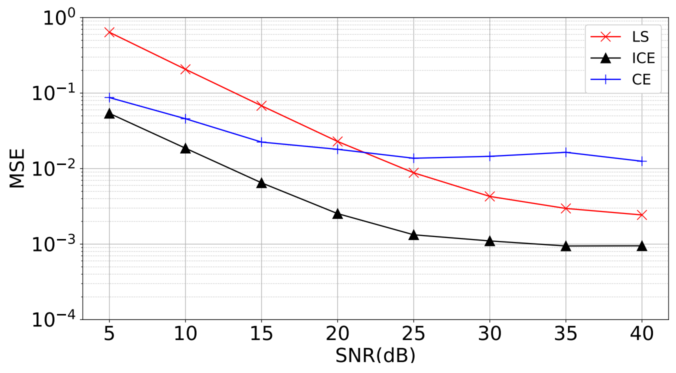
(b) Nonlinear scenario

Figure 5.14: Average BER for a CP-OFDM receiver with 64 subcarriers operating with block-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$. The scenario is linear in (a), that is, no clipping distortion is presented in the transmitter. The scenario is nonlinear in (b), that is, a clipping distortion with $C_R = 1.3$ is present in the transmitter.

The BER curves for the nonlinear scenario are shown in Figure 5.18 for the ITU Pedestrian A channel. All the BER curves are degraded due to the nonlinear distortion. Since this distortion is present at the transmitter side, the 'Exact' channel estimator does not know it. One can observe that 'ZP ICE+BFDE' with $K = L/2$ outperforms 'Exact+BFDE' using less redundancy. It unveils that the ICE subnet can handle the nonlinear distortion caused by the clipping process. In Figure 5.18b, we investigate how the resulting BER is affected when $\beta = 10$ data OFDM signals



(a) Linear scenario



(b) Nonlinear scenario

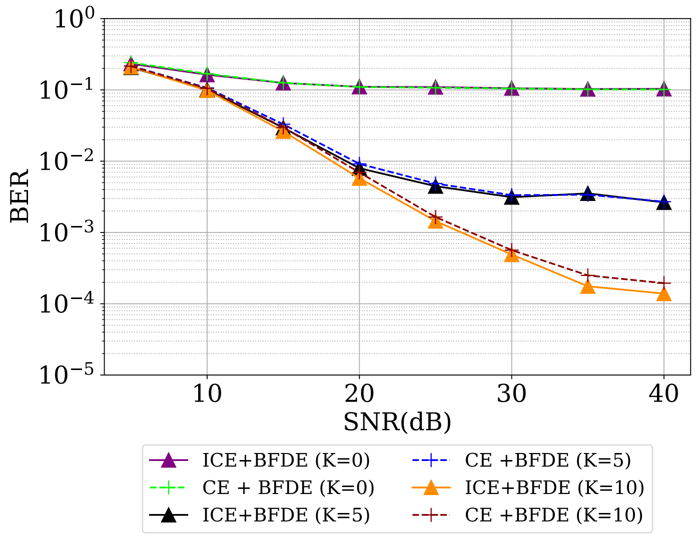
Figure 5.15: MSE between the true channel and the estimated channel for CP-OFDM with a block-type pilot arrangement and 64 subcarriers. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$. The scenario is linear in (a), that is, no clipping distortion is presented in the transmitter. The scenario is nonlinear in (b), that is, a clipping distortion with $C_R = 1.3$ is present in the transmitter.

are sequentially transmitted at a slowly-varying channel, in which β represents the number of transmitted data blocks. The overall performance is degraded, but the impact is less severe when SNR exceeds 20 dB.

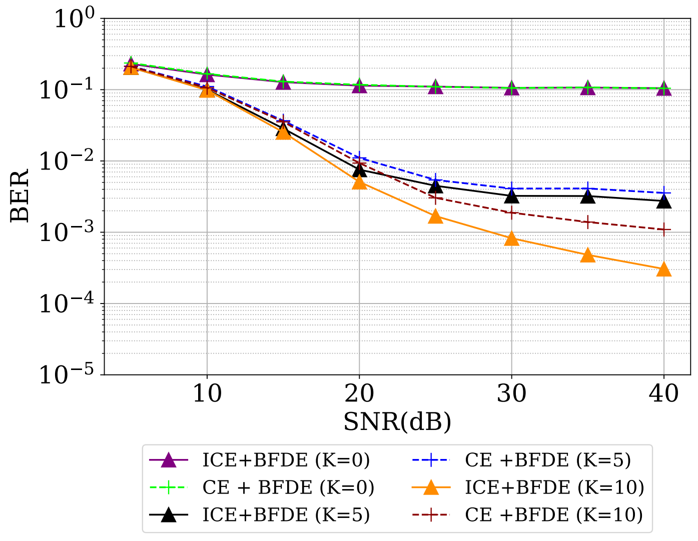
5.4.3 'ICE' solution for the comb-type pilot arrangement in CP-OFDM systems

In this section, we propose an ML-based OFDM receiver operating with insufficient redundancy, which means that the cyclic prefix length is shorter than the channel length $K < L$. We consider a comb-type arrangement for the transmitted pilots. The receiver is composed of the proposed improved channel estimator (ICE) network and the BFDE in equation (5.17), as depicted in Figure 5.11.

The ICE network aims to mitigate the nonlinear clipping distortion and obtain a better estimate for the data carriers, whereas the channel effects are inverted using



(a) Linear scenario

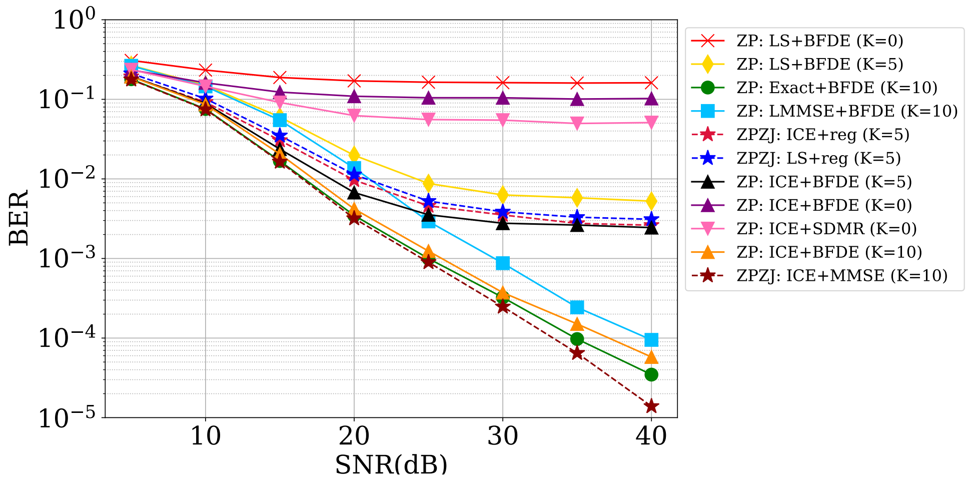


(b) Nonlinear scenario

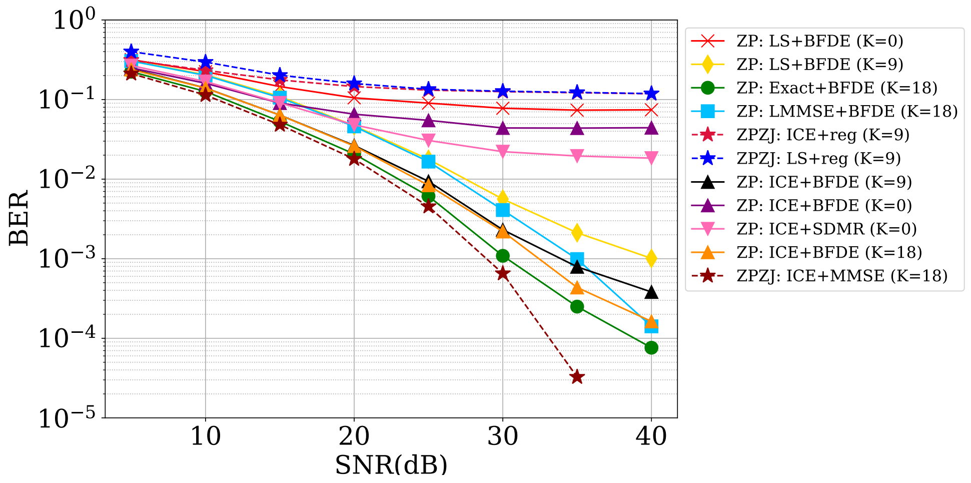
Figure 5.16: Comparison between ICE (proposed) and CE channel estimators for CP-OFDM with 64 subcarriers and a block-type pilot arrangement in terms of average BER. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$. The scenario is linear in (a), that is, no clipping distortion is presented in the transmitter. The scenario is nonlinear in (b), that is, a clipping distortion with $C_R = 1.3$ is present in the transmitter.

the BFDE. The simulations show that a simple network like ICE can perform well in transceiver configurations and environmental conditions. We extended the work of [130] and applied it to various situations, indicating that the approach is efficient with little additional complexity.

The real-valued block version of the complex LS channel estimate $\widehat{\boldsymbol{h}}_{LS} \in \mathbb{C}^{N\times 1}$ is the input of the ICE network. The real-valued block version of a vector is defined



(a) ITU Pedestrian A channel, $L = 10$



(b) WINNER II channel, $L = 18$

Figure 5.17: Average BER for a ZP-OFDM receivers with 64 subcarriers operating with block-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$ in (a), and the WINNER II channel model is used in (b). The scenario is linear, that is, no clipping distortion is presented in the transmitter.

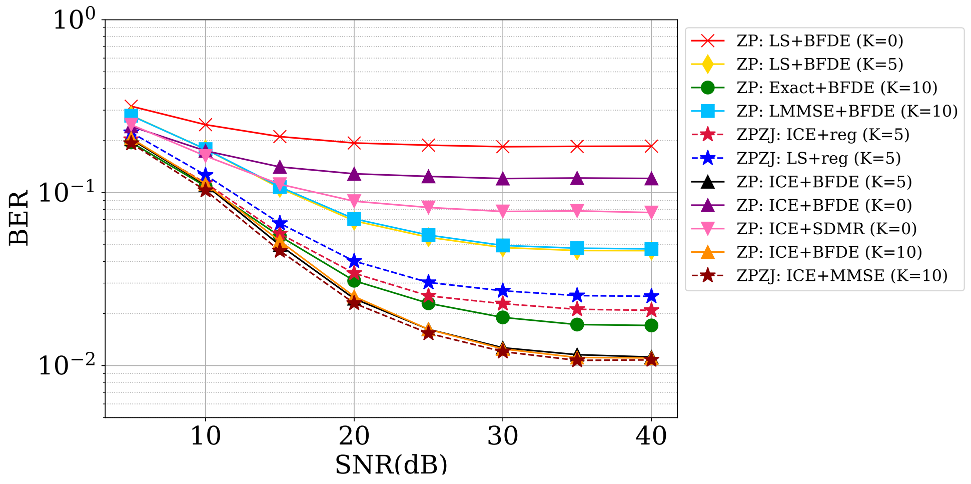
in equation (5.23). In the comb-type pilot arrangement, one needs to obtain CSI at the pilot subcarriers and use it to estimate the channel at the data subcarriers via interpolation. We use linear interpolation. With the channel estimate produced by the ICE subnet $\widehat{\boldsymbol{h}}_{\text{ICE}}$, we can obtain a frequency-domain equalizer solution,

$$y'_D(n) = \frac{y_D(n)}{\widehat{h}_{\text{ICE}}(n)} \quad \text{for} \quad n \in \mathbb{B}_D, \tag{5.30}$$

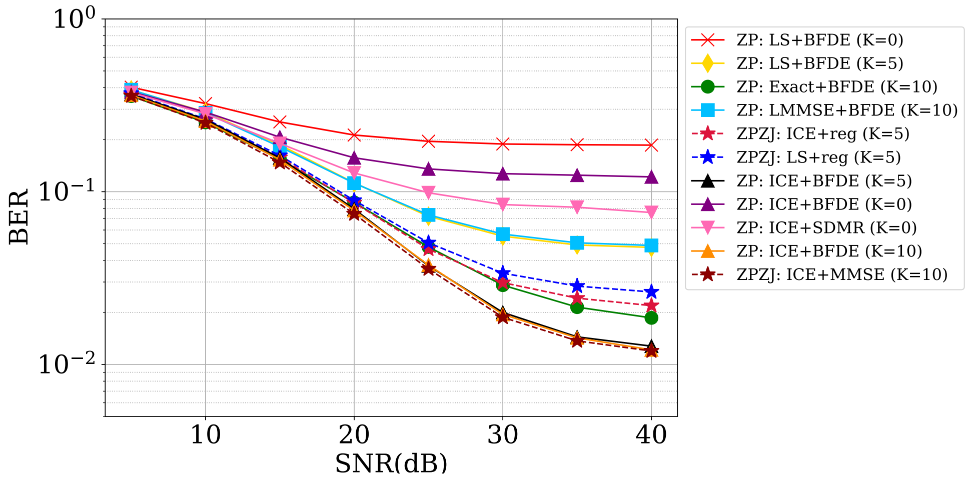
for the data subcarriers.

5.4.3.1 Simulation results considering 'ICE' solution

In this section, we consider the basic simulation setup in Section 5.3.



(a) Number of transmitted data blocks, $\beta = 1$



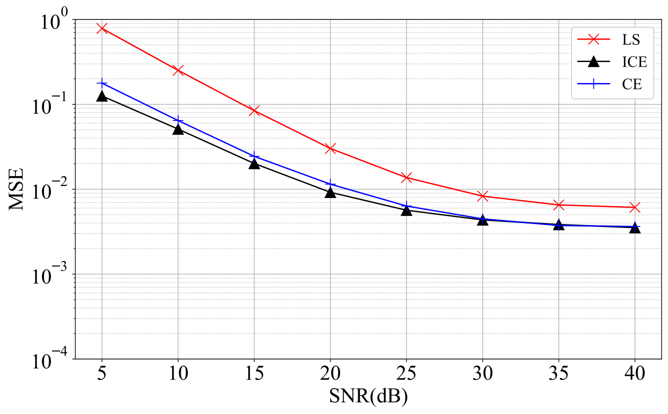
(b) Number of transmitted data blocks, $\beta = 10$

Figure 5.18: Average BER for a ZP-OFDM receivers with 64 subcarriers operating with block-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$. The scenario is nonlinear, that is, the clipping ratio is $C_R = 1.3$.

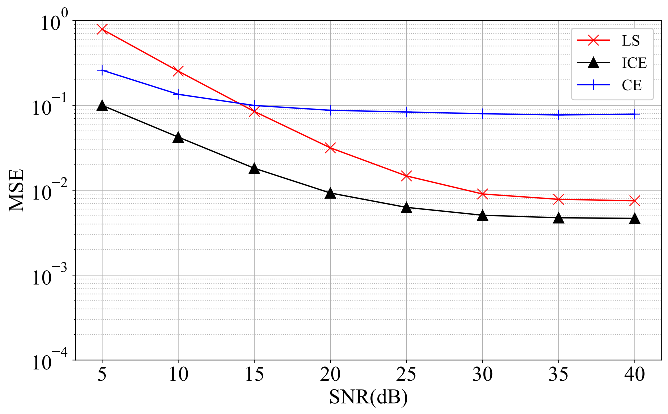
The tested methods employ a channel estimator followed by the BFDE equalizer in equation (5.17). For example, the name 'LS' represents the LS channel estimator in equation (5.13). 'Exact' means that the receiver has a perfect CSI. 'LMMSE' represents the LMMSE channel estimator in equation (5.14), where Perfect CSI is assumed to compute the correlation matrices. We consider the LMMSE solution as a standard method for comparison. 'ICE' is our proposed channel estimator. We use linear interpolation to estimate the channel at the data subcarriers. We save $N_p = 32$ subcarries to transmit the pilot symbols.

Figure 5.19 shows the MSE between the true and estimated channels using ICE, CE and LS channel estimators when $L = 10$. The ICE network outperforms the CE subnet, especially for the nonlinear scenario in Figure 5.19b.

The average BER for a CP-OFDM with the comb-type pilot arrangement is



(a) Linear scenario.



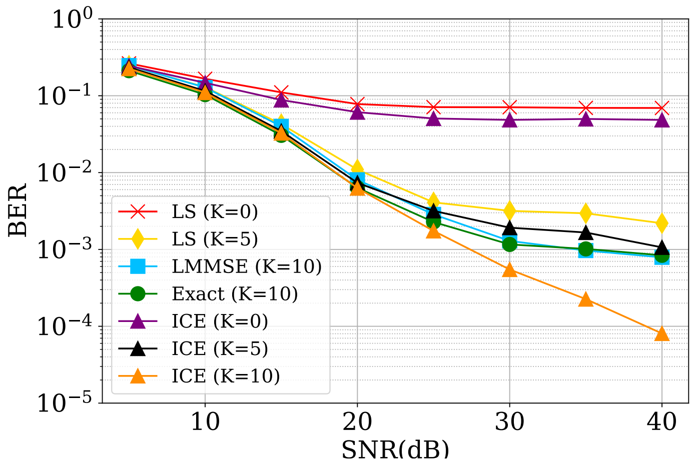
(b) Nonlinear scenario.

Figure 5.19: MSE between the true channel and the estimated channel for CP-OFDM with 64 subcarriers operating with comb-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$.

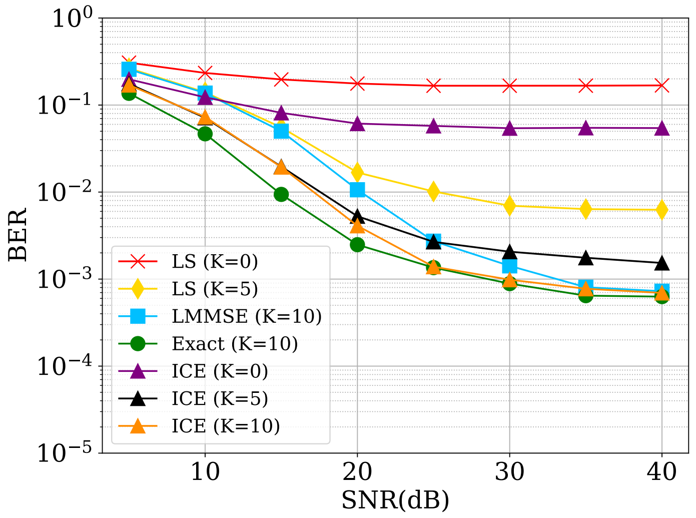
shown in Figure 5.20a for $L = 10$. The 'Exact' CP-OFDM receiver has perfect CSI only at the pilot tones, whereas the channel estimate at the data tones is obtained via interpolation. On the other hand, the ICE subnet has learned to estimate the channel at both the pilot and data tones. Therefore, when $K = L = 10$, the proposed 'ICE' outperforms the 'Exact' CP-OFDM receiver. When $K = L/2 = 5$, the proposed 'ICE' performance is quite close to the 'Exact' and 'LMMSE' receivers.

The nonlinear clipping distortion with $C_R = 3.5$ is considered in Figure 5.20b. One can note that the overall reception quality is reduced due to the unknown nonlinear distortion. In this case, the proposed 'ICE' with $K = L = 10$ performs quite close to the 'Exact' CP-OFDM receiver. When $K = L/2 = 5$, the proposed 'ICE' outperforms the 'LMMSE' CP-OFDM receiver for SNR < 30 dB.

Since the channel delay spread might vary in practice based on the propagation environment, we consider pedestrian channels with maximum delay spread



(a) Linear scenario.



(b) Nonlinear scenario.

Figure 5.20: Average BER for a CP-OFDM receiver with 64 subcarriers operating with comb-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$.

$L \in \{5, 10, 15\}$ during training in the following experiment. The example includes clipping in the training data with clipping ratio $C_R = 3.5$ in Figure 5.21. To simplify the evaluation, we only consider methods operating with CP length $K = 10$. As one can observe, the performance is similar to the case where a fixed maximum delay spread L is considered in Figure 5.20b. We then assume that the neural network addressed the issue.

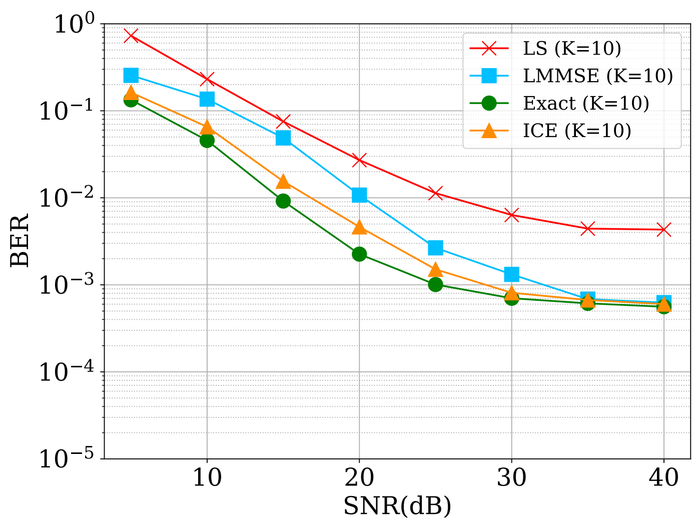


Figure 5.21: Average BER for a ZP-OFDM receivers with 64 subcarriers operating with comb-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is varied so that $L \in \{5, 10, 15\}$.

5.5 Redundancy length choice using deep reinforcement learning (RL)

In the previous section, we proposed two neural networks to be used when the employed redundancy is insufficient, and there is some sort of unknown nonlinearity at the transmitter side. Nevertheless, the redundancy length is prescribed at the beginning of the transmission and is constant during the whole communication. Therefore, motivated by the minimum redundancy analysis for ZPZJ-OFDM systems, we chose $K = L/2$.

Consider now an OFDM system where the transmitter should choose the CP length for each transmission so that the receiver can estimate the wireless communication channel with a particular target performance. This scenario reflects the problem of selecting the redundancy length when the channel order is unknown. We can formulate this problem using the reinforcement learning framework. As illustrated in Figure 5.22, the *agent* is the transmitter, the *action* is related to the CP length, the *environment* comprises the channel and the receiver, the *state* is the MSE between the estimated channel and the true channel, and the *reward* is related to the MSE.

Since the DQN usually considers discrete actions, we propose three actions. In the first one, the CP length does not change, the second action increases the CP length by one, and the third one decreases the CP length by one.

The reward function is a mechanism defined to specify how favorable the state

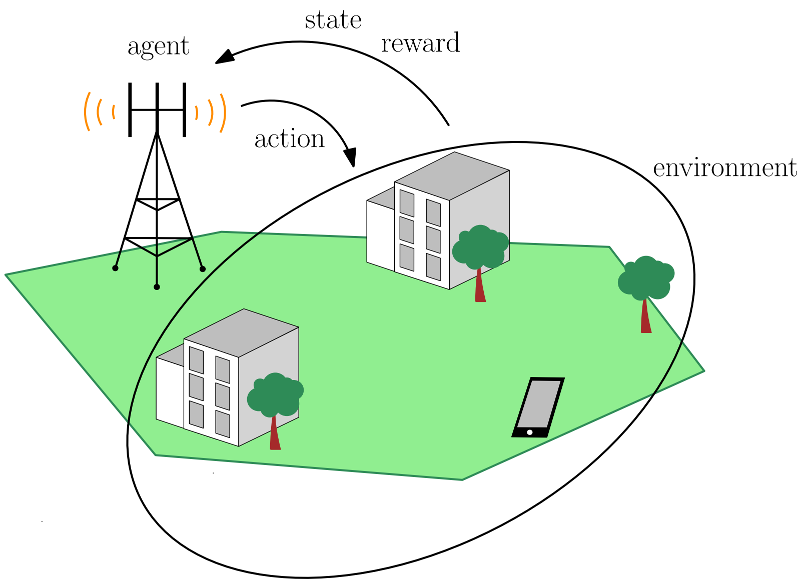


Figure 5.22: Reinforcement Learning applied to the considered OFDM system.

is in terms of a score. We utilize the reward function

$$r = 1 - (\nu/\nu_{\max})^{0.4}, \tag{5.31}$$

which utilizes a power of 0.4 to offer agents a smooth gradient of rewards as the MSE ν is getting closer to the maximum allowed MSE $\nu_{\max}$.

5.5.1 Simulation results

In this section, we consider the basic simulation setup in Section 5.3. To estimate the channel, we use the LS method defined in equation (5.13).

The MSE evolution between the channel estimate and the true channel during the online training is shown in Figure 5.23 when the SNR = 40 dB. We can observe that the algorithm converges in around 400 iterations to an MSE floor of 10^{-4}, and the correspondent obtained CP length is $K = 12$, as depicted in Figure 5.24. The rewards are aligned with the achieved MSE during training, as shown in Figure 5.25, in the sense that when the state, in this case, the MSE, is not desirable, the reward is low. As expected and shown in Figure 5.26, the exploration rate decreases as the RL algorithm is trained. This indicates that the RL algorithm explores the environment more at the beginning of training and exploits it more after 400 iterations, which is when the algorithm achieves convergence.

Considering that the channel order is $L = 10$, the obtained CP length in the last iteration is slightly higher than L. However, its value is close to $L = 10$. Therefore, the proposed DQN could be used in cases where the channel order is unknown or difficult to obtain.

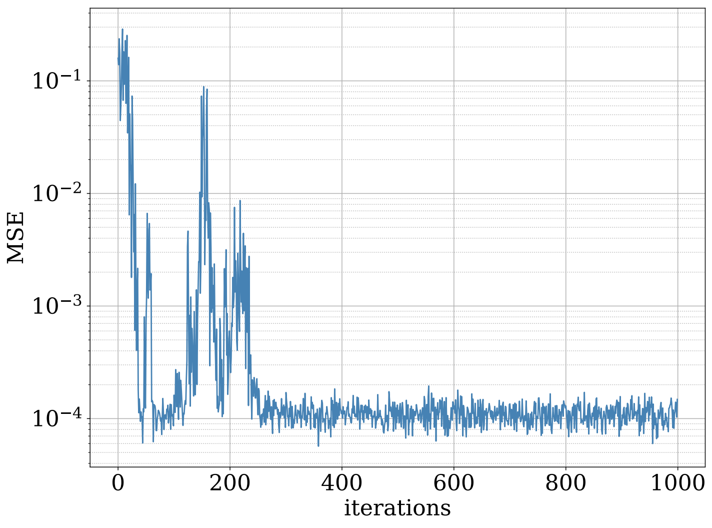


Figure 5.23: Evolution of the states during the deep RL online training. The state is the MSE between the channel estimate and the true channel.

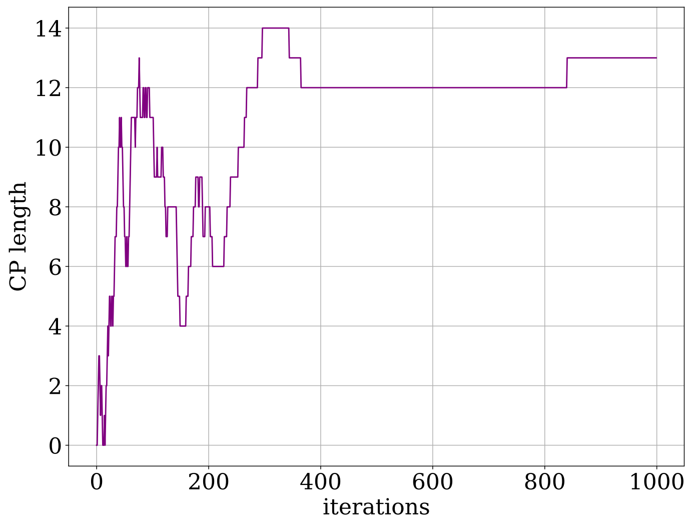


Figure 5.24: Evolution of the actions during the deep RL online training. The action is the chosen CP length for transmission.

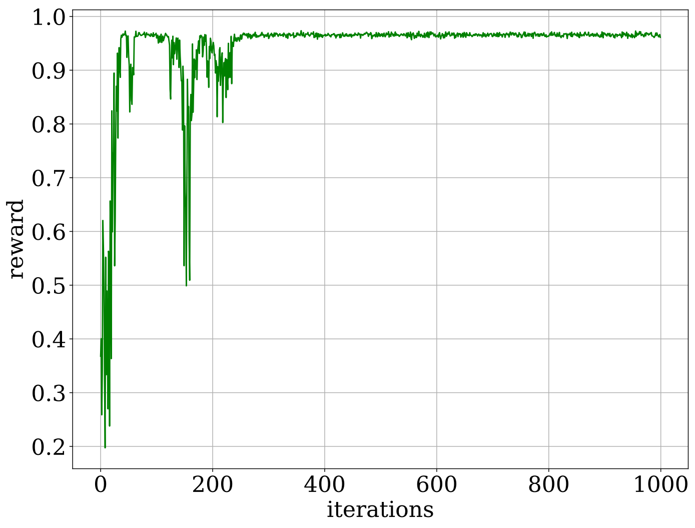


Figure 5.25: Evolution of the rewards during the deep RL online training. The reward function is defined in equation (5.31).

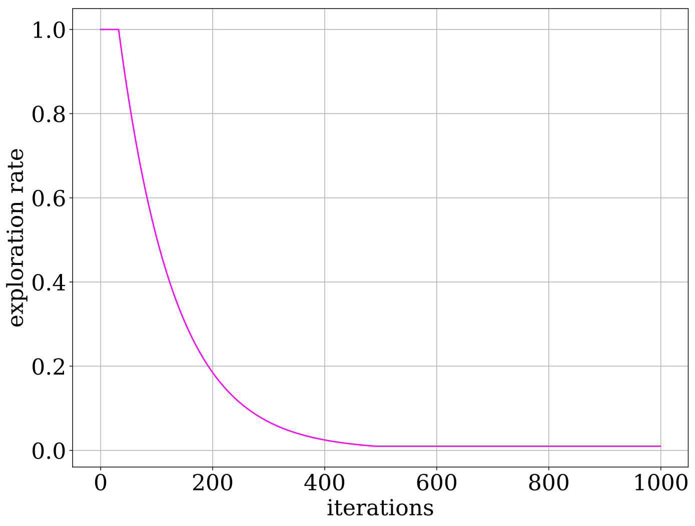


Figure 5.26: Evolution of the exploration rate during the deep RL online training.

5.6 Conclusions

In crucial applications where the spectrum efficiency is at stake in CP-OFDM and ZP-OFDM systems, the use of ML learning solutions, as preliminary discussed in this chapter, consists of a viable solution. It is shown that reduced BER performance can be achieved by utilizing OFDM transceivers using two-block NNs and employing insufficient redundancy. The first block is responsible for undoing the channel and nonlinear distortions. In contrast, the second block refines the received OFDM symbol contaminated by ISI and IBI due to insufficient redundancy. The simulations show that with linear channels and nonlinear power amplifiers at the transmitter, with insufficient redundancy, the ML-based solution provides performance benefits.

Moreover, when the channel order is unknown, choosing the redundancy length can be challenging. We then proposed a deep reinforcement learning framework to determine the amount of redundancy in a CP-OFDM system. The results show that the target MSE can be reached with the redundancy length that was automatically selected.

This chapter discussed issues with wireless OFDM systems related to channel order uncertainty, message distortions, and spectrum efficiency. Since the wireless OFDM system is open and vulnerable to malicious attacks, we address such a problem in the next chapter and propose a solution based on adversarial training.

Chapter 6

Robust machine learning for wireless OFDM systems

6.1 Introduction

The previous chapter addressed problems in wireless OFDM systems regarding spectrum efficiency, nonlinear distortions presented in the transmitted message, and channel order uncertainty by proposing machine learning-based solutions. However, as wireless OFDM systems are open, they can be targeted by jamming attacks. Since neural networks are known to be sensitive to adversarial attacks, this vulnerability grows when they are utilized to boost system performance. Therefore, this chapter proposes an adversarial training approach to cope with jamming attacks in wireless CP-OFDM systems.

As detailed in Chapter 5, OFDM is an efficient technique used in broadband wireless communications to combat multipath fading [14]. Pilot symbols are used in OFDM to obtain channel state information (CSI), which is often done using techniques like least squares (LS) and minimum mean square (MMSE). Accurate CSI is vital to ensure the receiver can perform channel equalization and properly detect the data symbols.

Due to the success of deep learning (DL) in many fields, DL-based channel estimation strategies have recently been presented to increase the accuracy of channel estimation [149]. For example, two neural networks were designed to refine the CSI accuracy and improve data detection in [130]. Convolutional neural networks (CNNs) are also commonly used to obtain CSI [150]. Nevertheless, a simple NN is sufficient to improve the LS channel estimates when insufficient redundancy is employed and nonlinear clipping distortion is present, as shown in Chapter 5.

As the wireless communication channel is open and exposed, the OFDM systems are prone to jamming attacks [151]. In wireless communications, jamming is defined as an intentional interfering signal hindering the transmission or distorting the legitimate signal. These harmful attacks have the potential to impair communications seriously. The risk increases when utilizing machine learning-based methods in wireless communications systems, as neural networks are known to be vulnerable to adversarial examples. As discussed in Chapter 3, adversarial examples are small and intended perturbations designed to fool neural networks. In this chapter, we explore pilot jamming attacks in wireless OFDM systems, as illustrated in Figure 6.1. The pilot tones are corrupted by an intentional interfering signal that aims to impair the channel estimation. Since we use a neural network to estimate the channel, the jamming attacks can be divided into two categories: conventional and smart. In this sense, the conventional jamming attack disregards the NN vulnerability to adversarial examples, whereas the smart jamming attack explores this vulnerability.

The OFDM system was previously detailed in Section 5.2 of Chapter 5; however, in Section 6.2, we provide a summary of the system for the reader convenience. We then formalize three different attacks, including conventional and smart jamming attacks in Section 6.3. In Section 6.4, we propose an adversarial training framework adapted to CSI acquisition in OFDM systems as a defense to pilot jamming attacks. We show some experimental results in terms of MSE to evaluate the robust OFDM receiver proposed in Section 6.5. The experimental results indicate that the proposed defense enhances CSI even in the absence of attacks. Section 6.6 includes some concluding remarks.

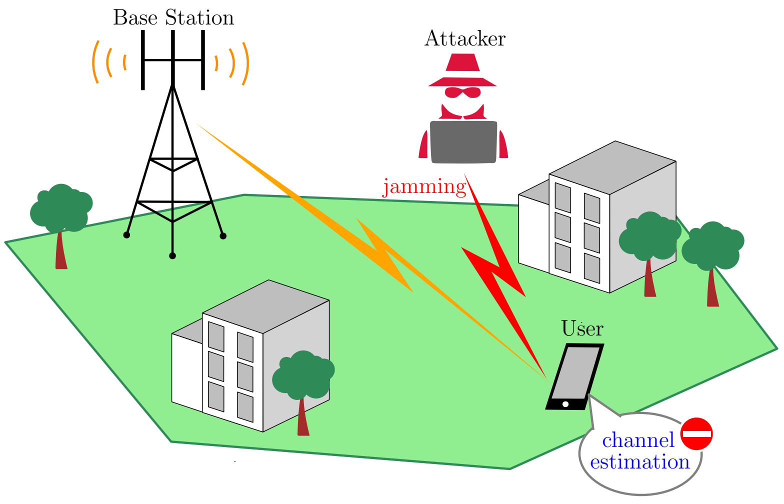


Figure 6.1: Illustration of jamming attack during the channel estimation of a wireless OFDM system.

6.2 ML-based OFDM channel estimation

In this section, we revisit the OFDM system presented in Chapter 5. As detailed in Section 5.2 of Chapter 5, the incoming data streams are modulated by using the M-ary quadrature amplitude modulation (M-QAM), resulting in the symbols $\boldsymbol{x}_P^{\mathrm{T}} \in \mathbb{C}^{1\times N}$. We consider a block-type arrangement where all subcarriers either contain pilots or data. This configuration is also considered in [125, 130] to simplify the simulations. The symbols are then converted to a parallel data stream $\boldsymbol{x}_P \in \mathbb{C}^{N\times 1}$. The N-point inverse fast Fourier transform (IFFT) is employed to convert the signal from the frequency domain $\boldsymbol{x}$ to the time domain $\mathbf{x}_P = \mathbf{W}_N^{\mathrm{H}}\boldsymbol{x}_P$, in which $\mathbf{W}_N$ is the unitary $N\times N$ discrete Fourier transform (DFT) matrix. After adding the K-length redundancy and performing the clipping operation, the OFDM signal

$$\mathbf{u}_c = \begin{cases} \mathbf{A}\mathbf{x}_P, & \text{if } |\mathbf{A}\mathbf{x}_P| < (C_R\sigma_u). \\ \frac{\mathbf{A}\mathbf{x}_P}{|\mathbf{A}\mathbf{x}_P|}(C_R\sigma_u), & \text{otherwise,} \end{cases} \tag{6.1}$$

is transmitted using N subcarriers. Low clipping ratio C_R leads to more severe nonlinear distortion, whereas $C_R \to \infty$ represents an absence of clipping noise. Matrix $\mathbf{A} \in \mathbb{C}^{S\times N}$ adds redundancy,

$$\mathbf{A} = \begin{bmatrix} \mathbf{0}_{K\times(N-K)} \; \mathbf{I}_K \\ \mathbf{I}_N \end{bmatrix}, \tag{6.2}$$

and σ_u is the root mean square (RMS) value of the OFDM signal [142].

The channel model between transmitter and receiver has the impulse response $\mathbf{h} = [\mathrm{h}(0)\ \ \mathrm{h}(1)\ \ \cdots \mathrm{h}(L)]^{\mathrm{T}}$. In the z-domain, the pseudo-circulant channel matrix is

$$\mathbf{H}(z) = \mathbf{H}_{\mathrm{ISI}} + z^{-1}\mathbf{H}_{\mathrm{IBI}} = \\ \begin{bmatrix} \mathrm{h}(0) & 0 & \cdots & 0 \\ \mathrm{h}(1) & \mathrm{h}(0) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{h}(L) & \mathrm{h}(L-1) & \cdots & 0 \\ 0 & \mathrm{h}(L) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \mathrm{h}(0) \end{bmatrix} + z^{-1} \begin{bmatrix} 0 & \mathrm{h}(L) & \cdots & \mathrm{h}(1) \\ 0 & \cdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \mathrm{h}(L) \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix}, \tag{6.3}$$

in which $\mathbf{H}_{\mathrm{ISI}}$ and $\mathbf{H}_{\mathrm{IBI}}$ respectively represent ISI and IBI effects produced by the wireless channel [15]. As in [125, 130], we assume that the channel remains approximately constant over the transmission of an OFDM frame. We also consider that only neighborhood blocks interfere with each other.

In the absence of noise, the received signal in the time-domain is

$$\mathbf{y}(k) = \mathbf{R}\mathbf{H}_{\text{ISI}}\mathbf{A}\mathbf{x}(k) + \mathbf{R}\mathbf{H}_{\text{IBI}}\mathbf{A}\mathbf{x}(k-1), \tag{6.4}$$

where the redundancy is removed by

$$\mathbf{R} = \left[\, \mathbf{0}_{N\times K} \; \mathbf{I}_N \,\right]. \tag{6.5}$$

If sufficient redundancy is inserted $K \geq L$, then

$$\mathbf{y}(k) = \mathbf{H}_c\mathbf{x}(k) = \mathbf{h}_N \circledast \mathbf{x}(k), \tag{6.6}$$

in which $\mathbf{h}_N$ is the first column of $\mathbf{H}_c$ and $\circledast$ is the circular convolution operator. We can employ the FFT to transform the circular convolution into component-wise multiplication using the circular convolution theorem. Hence, the LS channel estimate $\widehat{\boldsymbol{h}}_{\text{LS}} \in \mathbb{C}^{N\times 1}$ can be written for each subcarrier x

$$\widehat{h}_{\text{LS}}(n) = \frac{y_P(n)}{x_P(n)} \quad \text{for} \quad n = 1, \cdots N, \tag{6.7}$$

where $\boldsymbol{x}_P \in \mathbb{C}^{N\times 1}$ and $\boldsymbol{y}_P \in \mathbb{C}^{N\times 1}$ are the transmitted and received pilot signals in the frequency domain.

To obtain CSI, we can also employ the MMSE estimator with some practical implementation forms in [143].

If the OFDM system operates with insufficient redundancy $K < L$, a ZF solution is not possible for time-invariant transceivers. Aiming to undo the nonlinear clipping distortion while operating with insufficient redundancy, we utilize the improved channel estimator (ICE) net. The ICE net was described in details in Subsection 5.4.2.

6.3 Pilot jamming attacks in OFDM systems

Due to its exposed wireless nature, OFDM systems are vulnerable to jamming attacks. Interfering on pilot symbols used for channel estimation leads to noisy CSI, and hence performance degradation [152]. The vulnerability to pilot jamming increases when using a neural network to improve the channel estimation as NNs are susceptible to adversarial examples [153]. This section introduces three jamming attacks to perturb the OFDM channel estimation by increasing the MSE between the channel estimate and the actual channel. The first disregards the NN vulnerability to adversarial examples, whereas the remaining attacks explore it smartly.

6.3.0.1 Random jamming attack

We consider a conventional jamming attack in which the jamming signal is AWGN with variance σ_n^2 [154]. In this case, the jammer can be located anywhere between the transmitter and the receiver. The random jamming attack does not intend to fool the NN specifically. Instead, it aims to increase the MSE between the channel estimate and the true channel for both channel estimators.

6.3.0.2 Worst-case scenario jamming attack

We can characterize a worst-case scenario by assuming that the signal processing chain has been compromised [155]. In this case, the attacker is located inside the receiver, and it can directly perturb the LS channel estimate by adding a perturbation $\boldsymbol{\eta}$, resulting in the adversarial example

$$\boldsymbol{h}_{\text{adv}} = \widehat{\boldsymbol{h}}_{\text{LS}} + \boldsymbol{\eta}. \tag{6.8}$$

The worst-case scenario jamming attack intends to fool only the NN by increasing the MSE between the channel estimate and the true channel. It has access to the model weights to craft $\boldsymbol{\eta}$ so that the MSE between the channel estimate and the actual channel is maximally increased without amplifying $\boldsymbol{\eta}$.

6.3.0.3 Eavesdropping-assisted jamming attack

Looking for a smart attack that could be used in practice, we propose the eavesdropping-assisted jamming attack. In this attack, the jammer is located near the transmitter, and the jamming signal is based on the adversarial example defined in equation (6.8). Nevertheless, the jammer must know the channel estimate to build the adversarial perturbation to generate the jamming signal. To do so, a pair of eavesdrop sensors can be employed to estimate the channel and send this information to the actual jammer, as illustrated in Figure 6.2. This situation is plausible in internet of things (IoT) environments.

Since the jammer needs to transmit a signal $\mathbf{s}_J$ so that it can be characterized as an attack, we can rewrite equation (6.8) for each subcarrier n

$$h_{\text{adv}}(n) = \frac{y_P(n)}{x_P(n)} + \eta(n), \tag{6.9}$$

and find a perturbation $\eta(n)$ dependent on $s_J(n)$. To do so, let us express the overall received signal at the receiver as

$$\mathbf{y}_{\text{ov}}(k) = \mathbf{y}_P(k) + \mathbf{y}_J(k) \tag{6.10}$$

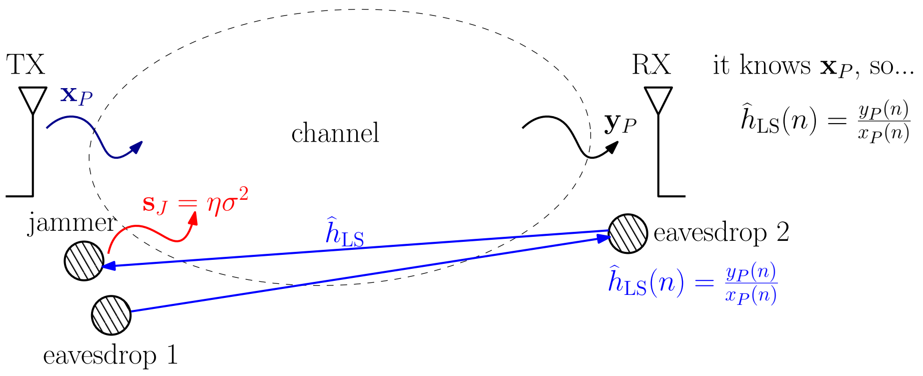

TX
$\mathbf{x}_P$
channel
RX
$\mathbf{y}_P$
it knows $\mathbf{x}_P$, so...
$\hat{h}_{\text{LS}}(n) = \frac{y_P(n)}{x_P(n)}$
$\mathbf{s}_J = \eta\sigma^2$
jammer
$\hat{h}_{\text{LS}}$
eavesdrop 2
$\hat{h}_{\text{LS}}(n) = \frac{y_P(n)}{x_P(n)}$
eavesdrop 1

Figure 6.2: Illustration of jamming attack based on adversarial samples.

which is composed of the original message $\mathbf{y}_P$ and the jamming signal $\mathbf{y}_J$ during the pilot phase. In the time domain, the received original signal can be described as

$$\mathbf{y}_P(k) = \mathbf{H}_{\text{ISI}}\mathbf{u}(k) + \mathbf{H}_{\text{IBI}}\mathbf{u}(k-1) + \mathbf{v}(k), \tag{6.11}$$

whereas the received jamming signal is

$$\mathbf{y}_J(k) = \mathbf{H}_{\text{ISI}}\mathbf{u}_J(k) + \mathbf{H}_{\text{IBI}}\mathbf{u}_J(k-1) + \mathbf{v}'(k). \tag{6.12}$$

In the absence of noise and after removing the redundancy, we obtain

$$\begin{aligned}\mathbf{y}_{\text{ov}}(k) = {} & \mathbf{R}\mathbf{H}_{\text{ISI}}\mathbf{A}\mathbf{x}(k) + \mathbf{R}\mathbf{H}_{\text{IBI}}\mathbf{A}\mathbf{x}(k-1) \\ & + \mathbf{R}\mathbf{H}_{\text{ISI}}\mathbf{A}\mathbf{s}_J(k) + \mathbf{R}\mathbf{H}_{\text{IBI}}\mathbf{A}\mathbf{s}_J(k-1),\end{aligned} \tag{6.13}$$

where $\mathbf{A}$ is the matrix that includes the redundancy, defined in equation (6.2).

If the redundancy length is adequate, the IBI is eliminated $\mathbf{R}\mathbf{H}_{\text{IBI}}\mathbf{A} = \mathbf{0}$ and $\mathbf{R}\mathbf{H}_{\text{ISI}}\mathbf{A} = \mathbf{H}_{\text{c}}$ is a circulant matrix. Therefore, the matrix multiplication is equivalent to a circular convolution,

$$\mathbf{y}_{\text{ov}}(k) = \mathbf{H}_{\text{c}}\mathbf{x}(k) + \mathbf{H}_{\text{c}}\mathbf{s}_J(k) = \mathbf{h}_N \circledast [\mathbf{x}(k) + \mathbf{s}_J(k)]. \tag{6.14}$$

In the frequency domain, we can then use the LS method to estimate the channel at each subcarrier n

$$\begin{aligned}\hat{h}'_{\text{LS}}(n) = \frac{y_{\text{ov}}(n)}{x_P(n)} &= \frac{y_P(n)}{x_P(n)} + \frac{h_N(n)s_J(n)}{x_P(n)} \\ &= \frac{y_P(n)}{x_P(n)} + \eta(n).\end{aligned} \tag{6.15}$$

Observe that equation (6.15) is equivalent to the adversarial examples built in equation (6.9). From equation (6.15), the jamming signal can then be expressed as

$$s_J(n) = \eta(n)\frac{x_P(n)}{h_N(n)}. \tag{6.16}$$

We can then obtain an approximate expression for the jamming signal

$$s'_J(n) = \frac{\eta(n)}{||\eta(n)||}E\left[\left(\eta(n)\frac{x_P(n)}{h_N(n)}\right)^2\right] = \eta'(n)\sigma^2, \tag{6.17}$$

so that the signal has the adversarial perturbation $\eta(n)$ waveform, and its power is greater than σ^2.

The adversarial perturbation $\boldsymbol{\eta}$ can be computed using the Fast Gradient Sign Method (FGSM) introduced in equation (3.15), and it is repeated as follows for the reader convenience

$$\boldsymbol{\eta} = \epsilon\text{sign}(\nabla_{\boldsymbol{h}_{\text{LS}}}\mathcal{L}(\boldsymbol{\Theta}, \boldsymbol{h}_{\text{LS}}, \boldsymbol{h}_{\text{N}})), \tag{6.18}$$

in which sign$(\cdot)$ is the sign function, and $\nabla_{\boldsymbol{h}_{\text{LS}}}\mathcal{L}(\boldsymbol{\Theta}, \boldsymbol{h}_{\text{LS}}, \boldsymbol{h}_{\text{N}})$ is the loss gradient with respect to $\boldsymbol{h}_{\text{LS}}$.

6.4 Robust OFDM channel estimation

In this section, we consider that the OFDM receiver is under attack by a jamming signal. The jamming signal aims at perturbing the channel estimation at the receiver. Our proposed defense consists of training the ICE net with adversarial examples to cope with the possible jamming attacks.

We aim at finding the perturbation $\boldsymbol{\eta}$ which increases the MSE between the channel estimate obtained by the ICE net and the target channel $\mathcal{L}(f_{\boldsymbol{\theta}}(\widehat{\boldsymbol{h}}_{\text{LS}}+\boldsymbol{\eta}), \boldsymbol{h}_{\text{N}})$, but also keeps the MSE between the channel estimate obtained with the LS method and the target channel $\mathcal{L}(\widehat{\boldsymbol{h}}_{\text{LS}}+\boldsymbol{\eta}, \boldsymbol{h}_{\text{N}})$ barely affected. To meet these requirements, we propose the following modified optimization problem

$$\begin{aligned} \text{min}_{\boldsymbol{\theta}}\frac{1}{|\mathcal{D}|}\sum_{\widehat{\boldsymbol{h}}_{\text{LS}},\boldsymbol{h}_{\text{N}}\in\mathcal{D}} \text{max}_{\boldsymbol{\eta}}\ &(\mathcal{L}(f_{\boldsymbol{\theta}}(\widehat{\boldsymbol{h}}_{\text{LS}}+\boldsymbol{\eta}), \boldsymbol{h}_{\text{N}}) \\ &-\lambda\mathcal{L}(\widehat{\boldsymbol{h}}_{\text{LS}}+\boldsymbol{\eta}, \boldsymbol{h}_{\text{N}})) \\ \text{s.t. } &||\boldsymbol{\eta}||_p \leq \epsilon_0(1-r)^{\frac{SNR}{5}-1}, \end{aligned} \tag{6.19}$$

in which λ is a scaling factor that balances the importance of the two requirements. The initial perturbation constraint is ϵ_0, and r is the decay factor that controls how the perturbation decreases. Since less perturbation is required to increase the overall loss function for high SNR values, the constraint in the perturbation vector $\boldsymbol{\eta}$ is now dependent on the SNR, decaying with a rate r.

By setting $\lambda = 0$ and $r = 0$ in equation (6.19), we end up with the original optimization problem for classification problems. In this case, the inner maximization problem is responsible for finding the perturbation $\boldsymbol{\eta}$ that increases the loss $\mathcal{L}(f_{\boldsymbol{\theta}}(\widehat{\boldsymbol{h}}_{\mathrm{LS}} + \boldsymbol{\eta}),\boldsymbol{h}_{\mathrm{N}})$ in an attempt to change the predicted class. In our case, since the channel estimation is a regression problem; we are only interested in increasing the loss function, which is the MSE between the channel estimate obtained with the NN and the target channel.

Nevertheless, the adversarial perturbation should also guarantee the similarity between the adversarial sample and the original sample. In this regard, the optimization constraints ensure that the distance between the adversarial and original examples is less than ϵ under a particular norm, $||\boldsymbol{\eta}||_p \leq \epsilon$. The norms aim to quantify how imperceptible an adversarial example is to humans for computer vision tasks. Some examples of norms are the l_0 norm, l_2 norm, and l_∞. Since we estimate the channel with the LS method before using the obtained estimate as input for the ICE net, in our case, the LS estimator plays the role of the human eyes. Therefore, we aim to produce adversarial examples that fool only the NN, whereas the LS method is not deceived and will barely notice the attack. Here, deceiving or fooling means increasing the MSE between the channel estimate and the target channel. In this way, we can account for the similarity between the adversarial and clean samples.

6.4.1 Fast Gradient Sign Method (FGSM) to solve the inner maximization problem

The inner maximization problem is then solved by considering the Fast Gradient Sign Method (FGSM) attack [72]. FGSM generates adversarial examples by modifying the input towards the direction where the overall loss

$$\mathcal{L}_{\mathrm{o}}(\boldsymbol{\theta}, \widehat{\boldsymbol{h}}_{\mathrm{LS}}, \boldsymbol{h}_{\mathrm{N}}) = \mathcal{L}(f_{\boldsymbol{\theta}}(\widehat{\boldsymbol{h}}_{\mathrm{LS}}), \boldsymbol{h}_{\mathrm{N}}) - \lambda \mathcal{L}(\widehat{\boldsymbol{h}}_{\mathrm{LS}}, \boldsymbol{h}_{\mathrm{N}})) \tag{6.20}$$

increases. For the original optimization problem, $\lambda = 1$ in equation (6.20). As a result, the adversarial example is

$$\boldsymbol{h}_{\mathrm{adv}} = \widehat{\boldsymbol{h}}_{\mathrm{LS}} + \epsilon \mathrm{sign}(\nabla_{\widehat{\boldsymbol{h}}_{\mathrm{LS}}} \mathcal{L}_{\mathrm{o}}(\boldsymbol{\theta}, \widehat{\boldsymbol{h}}_{\mathrm{LS}}, \boldsymbol{h}_{\mathrm{N}})), \tag{6.21}$$

where $\epsilon = \epsilon_0 (1-r)^{\frac{SNR}{5}-1}$ for the modified optimization problem, $\mathrm{sign}(\cdot)$ is the sign function, and $\nabla_{\widehat{\boldsymbol{h}}_{\mathrm{LS}}} \mathcal{L}_{\mathrm{o}}(\boldsymbol{\theta}, \widehat{\boldsymbol{h}}_{\mathrm{LS}}, \boldsymbol{h}_{\mathrm{N}})$ is the loss gradient with respect to the input $\widehat{\boldsymbol{h}}_{\mathrm{LS}}$.

6.4.2 Outer minimization problem

With the inner maximization problem addressed, the outer minimization problem in equation (6.19) is then solved to find the model parameters that minimize the loss on the generated adversarial examples. The original dataset $\mathcal{D}$ is split into small batches $\mathcal{B}$ and stochastic gradient descent (SGD) is employed to update the model parameters

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \mu \frac{1}{|\mathcal{B}|} \sum_{\widehat{\boldsymbol{h}}_{\mathrm{LS}}, \boldsymbol{h}_{\mathrm{N}} \in \mathcal{B}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{o}}(\boldsymbol{\theta}, \widehat{\boldsymbol{h}}_{\mathrm{LS}} + \boldsymbol{\eta}^*, \boldsymbol{h}_{\mathrm{N}}), \tag{6.22}$$

where the gradient is evaluated at the maximum point $\boldsymbol{\eta}^*$ found in the inner maximization problem, thanks to Danskin's theorem [83].

6.5 Simulation results

In this section, we evaluate the proposed defense based on adversarial training via some simulation results, following the setup described in Subsection 6.5.1. In Subsection 6.5.2, we consider the original optimization problem when performing the defense and adversarial attacks, whereas Subsection 6.5.3 provides the results when the proposed optimization problem is employed to perform the defense and the adversarial attacks.

6.5.1 Simulation Setup

In this section, we consider the basic simulation setup in Section 5.3. The ICE net is trained for each signal-to-noise ratio (SNR) with $K = 5$. The CP-OFDM system has 64 subcarriers operating in a block-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model with $L = 10$.

6.5.2 Scenario 1: original optimization problem

We first consider the optimization problem in equation (6.19) with $\lambda = 0$ and $r = 0$ to perform the defense and to craft the adversarial attacks. The mean square error (MSE) between the channel estimate and the actual channel is presented as a function of the SNR for random, worst-case and eavesdropping-assisted jamming attacks in Figures 6.3, 6.4 and 6.5, respectively. The dashed lines represent a system free of attacks, whereas the solid lines indicate the system is under attack. NN standard is the ICE net trained only with clean samples, NN robust stands for the ICE net trained with adversarial examples, and LS is the basic method for estimating the channel without a neural network.

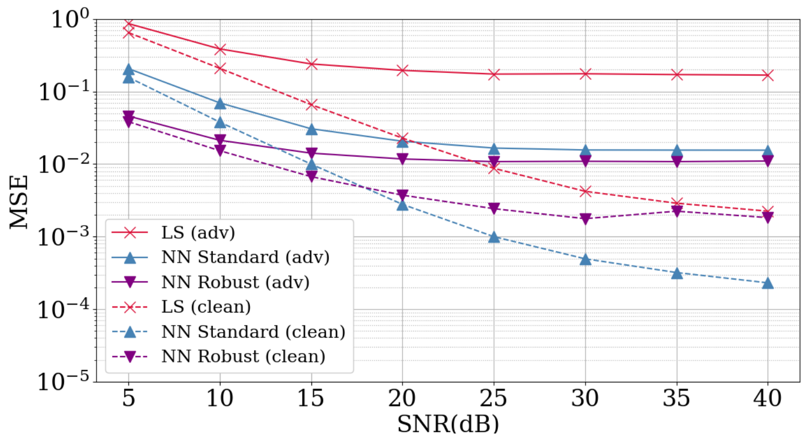


Figure 6.3: MSE results when the OFDM receiver is under the random jamming attack. The defense is based on the optimization problem in equation (6.19) with $\lambda = 0$ and $r = 0$. The CP-OFDM system has 64 subcarriers and operates with a block-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$. The CP lenght is $K = 5$. The scenario is nonlinear, that is, a clipping distortion with $C_R = 1.3$ is present in the transmitter.

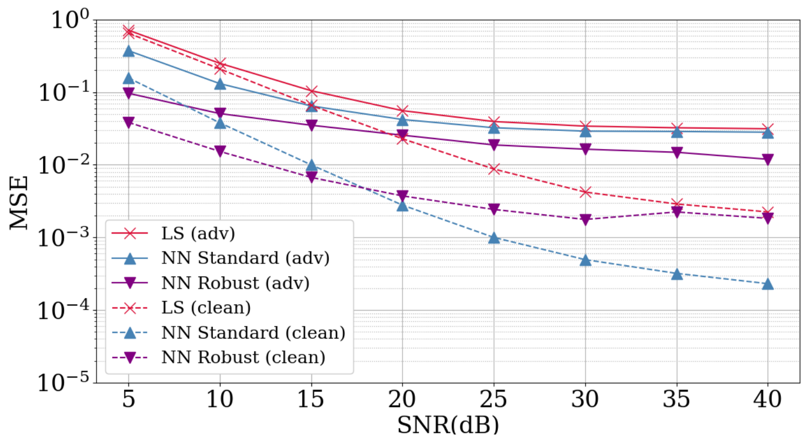


Figure 6.4: MSE results when the OFDM receiver is under the worst-case jamming attack. The defense is based on the optimization problem in equation (6.19) with $\lambda = 0$ and $r = 0$. The CP-OFDM system has 64 subcarriers and operates with a block-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$. The CP lenght is $K = 5$. The scenario is nonlinear, that is, a clipping distortion with $C_R = 1.3$ is present in the transmitter.

When the original optimization problem is considered, the impairment caused by the worst-case jamming attack in the standard trained network is the most severe among the considered attacks. This can be noticed by the gap between the solid and dashed blue curves in Figure 6.4. Although the worst-case jamming attack impairs the LS method less than the other attacks, there is still a significant gap between the

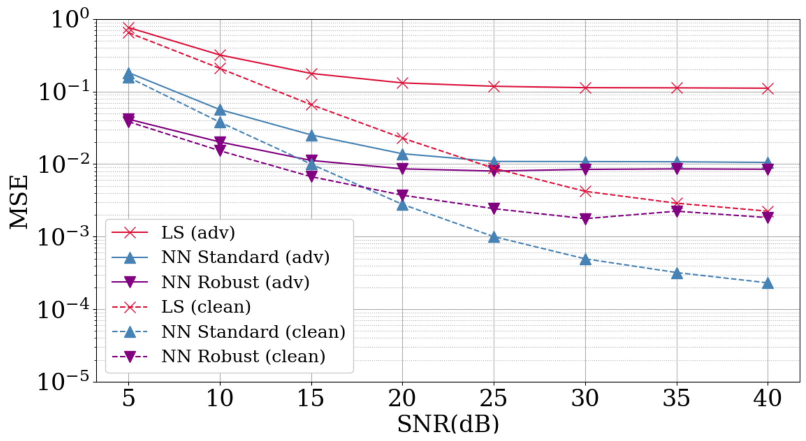


Figure 6.5: MSE results when the OFDM receiver is under the eavesdropping-assisted jamming attack. The defense is based on the optimization problem in equation (6.19) with $\lambda = 0$ and $r = 0$. The CP-OFDM system has 64 subcarriers and operates with a block-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$. The CP lenght is $K = 5$. The scenario is nonlinear, that is, a clipping distortion with $C_R = 1.3$ is present in the transmitter.

solid and dashed red curves in Figure 6.4. This is reasonable since the loss function does not consider the similarity constraints between the adversarial example and the original sample. Therefore, the LS performance is affected in this case. As shown in Figures 6.3 and 6.5, both the random and the eavesdropping-assisted jamming attacks yield similar results. This unveils that we are possibly adding an overpowered adversarial perturbation; that is, the adversarial perturbation is so high that it damages not only the NN results but also the LS results significantly as in the random case.

6.5.3 Scenario 2: proposed optimization problem

Finally, we consider the optimization problem proposed in equation (6.19) with $\lambda = 0.2$ and $r = 0.2$ to perform the defense and to craft the random, worst-case and eavesdropping-assisted jamming attacks in Figures 6.6, 6.7 and 6.8, respectively. In this case, the NN robust outperforms or performs quite similarly to the NN standard for clean samples (dashed lines). This shows that we can benefit from adversarial training even when the system is free of jamming attacks. As shown in Figure 6.7, the impairment caused by the worst-case jamming attack in the standard trained network is no longer the most severe among the attacks. Now we are not only interested in increasing the MSE when using the NN but also in minimally affecting the MSE when using the LS method. To perform a fair comparison, we varied the power of the jamming signal as in equation (6.19) when performing the random attack in Figure 6.6. Even though the impact on the MSE is more severe for the random attack, the gap between the solid and dashed red curves is larger if we compare it with the worst-case and the eavesdropping-assisted jamming attacks. Unlike the previous scenario, the eavesdropping-assisted jamming attack is more similar to the worst-case jamming attack. This encourages using the eavesdropping-assisted scheme to craft jamming attacks that try to fool the NN without impacting the LS method too much in practice, as the random attack does.

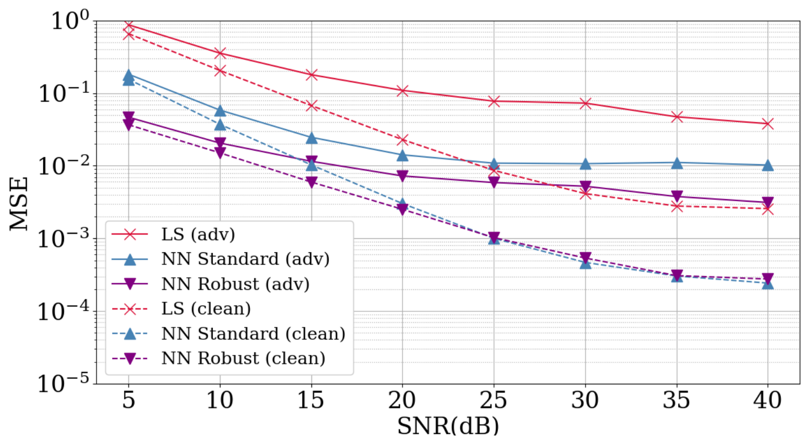


Figure 6.6: MSE results when the OFDM receiver is under the random jamming attack. The defense is based on the optimization problem in equation (6.19) with $\lambda = 0.2$ and $r = 0.2$. The CP-OFDM system has 64 subcarriers and operates with a block-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$. The CP lenght is $K = 5$. The scenario is nonlinear, that is, a clipping distortion with $C_R = 1.3$ is present in the transmitter.

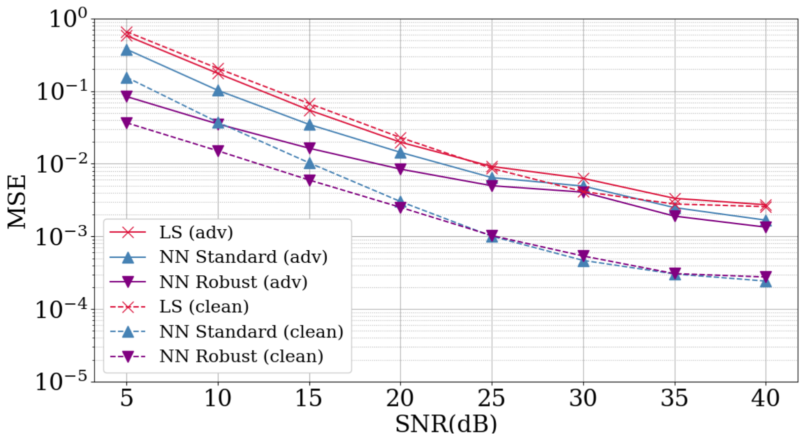


Figure 6.7: MSE results when the OFDM receiver is under the worst-case jamming attack. The defense is based on the optimization problem in equation (6.19) with $\lambda = 0.2$ and $r = 0.2$. The CP-OFDM system has 64 subcarriers and operates with a block-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$. The CP lenght is $K = 5$. The scenario is nonlinear, that is, a clipping distortion with $C_R = 1.3$ is present in the transmitter.

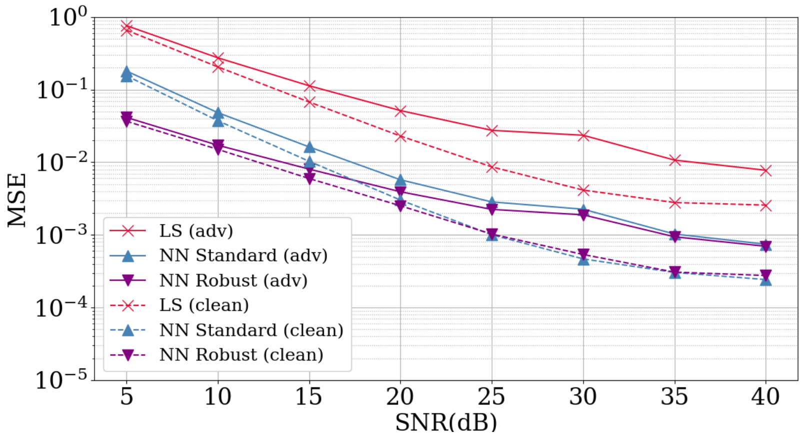


Figure 6.8: MSE results when the OFDM receiver is under the eavesdropping-assisted jamming attack. The defense is based on the optimization problem in equation (6.19) with $\lambda = 0.2$ and $r = 0.2$. The CP-OFDM system has 64 subcarriers and operates with a block-type pilot arrangement. 16-QAM symbols are transmitted through a channel modeled using a pedestrian model in which the channel order is $L = 10$. The CP lenght is $K = 5$. The scenario is nonlinear, that is, a clipping distortion with $C_R = 1.3$ is present in the transmitter.

6.6 Conclusions

In this chapter, we proposed a defense based on adversarial training to cope with jamming attacks during the channel estimation of a wireless CP-OFDM system. We considered three types of attacks to evaluate the proposed defense method. The simulations show that adversarial training can enhance the performance of ML-based channel estimators in CP-OFDM systems even when no jamming attack is present.

Chapter 7

Conclusions

7.1 Final remarks

Wireless communications technology has advanced significantly during the past few years. First, the wireless networks transitioned from analog to digital and began to provide basic web applications, and subsequently, new applications were available, demanding higher data rates. As wireless technologies evolve into 6G systems, the demand for higher data rates keeps growing. The possible leading solutions for beyond 5G and 6G communication systems include massive MIMO and OFDM. However, despite their advantages, massive MIMO and OFDM face implementation challenges. Through this work, we proposed solutions based on greedy algorithms and ML techniques to both techniques' implementation issues.

Due to the high cost of BS fabrication, massive MIMO systems are challenging to implement. Since selecting a subset of the transmitting antennas is an alternative to cope with the high cost of the BS, we proposed three low-complexity antenna selection algorithms based on the matching pursuit technique. We started by formulating a channel–level antenna selection algorithm in which we obtain the subset of antennas based only on the current channel estimate. We then proposed three symbol–level antenna selection algorithms: ZFSL-MPAS, SL-MPPAS, and SL-MPGBPPAS. ZFSL-MPAS uses the channel estimate and the messages to be transmitted to select the antennas as a symbol– level algorithm. Still, ZFSL-MPAS applies ZP precoding after obtaining the subset of antennas. Besides, SL-MPPAS and SL-MPGBPPAS do not require performing an external precoding scheme after selecting the antennas. SL-MPPAS and SL-MPGBPPAS consider precoding while selecting the antennas, merging the two operations, and lowering the computational complexity because no matrix inversion is required. Furthermore, since SL-MPGBPPAS generates a message with quantized components, it can alleviate the amplifier linearity demands and further reduce the BS fabrication cost. The proposed algorithms were evaluated and compared to the state-of-the-art using the BER and MSE. The simulation results indicate that short coherence times make symbol-level algorithms appealing in practice.

OFDM is another promising technique for 6G systems. Nevertheless, OFDM has some drawbacks, including spectrum waste brought by the overhead of the redundancy length and nonlinear distortion generated by methods to mitigate the peak-to-average power ratio (PAPR). Additionally, selecting the appropriate amount of redundancy can be challenging when the channel order is unknown. We, therefore, presented ML-based strategies to address these problems. First, we proposed ICE and SDMR neural networks to improve the OFDM system performance when nonlinearity from PAPR reduction techniques is present at the transmitter, and the OFDM symbol has insufficient redundancy. ICE net aims to refine the LS channel estimate, whereas SDMR is in charge of enhancing the detected symbol. Next, we applied the introduced NNs in OFDM systems with cyclic and zero-prefix under the block and comb-type pilot arrangements. The simulations show that the proposed OFDM receivers can improve the BER performance in the considered scenarios. We then proposed a deep reinforcement learning framework to select the amount of redundancy in a CP-OFDM system based on the current MSE between the actual and the estimated channel response. The automatically chosen redundancy length is sufficient to yield the MSE target previously prescribed.

The considered OFDM systems are sensitive to jamming attacks such as the pilot jamming that aims to harm the channel estimation. Even though NNs can be employed to enhance channel estimates, the risk is increased because NNs can be deceived by adversarial examples. Finally, we proposed an adversarial training approach to improve the channel estimation. The simulations demonstrate that, even in the absence of a jamming attack, adversarial training can improve the performance of ML-based channel estimators in CP-OFDM systems.

7.2 Future work

In this work, we proposed a DS approach in which the data samples are selected at the batch level in order to improve the classification accuracy. We also used the DS strategy to improve adversarial training's robust-accuracy tradeoff. We believe that the NNs utilized to address the wireless communications issues in our study can also benefit from this performance enhancement. Therefore, future work concerns applying the proposed DS strategy to the training process of the NNs used to improve the channel estimation (ICE net) and the detected symbol (SDMR net). To balance the proportion of clean and adversarial samples and to gain a deeper understanding of robustness in the context of wireless networks, the DS approach can also be used when training the ICE net with adversarial examples.

The adversarial training framework introduced in this work can also be extended to the context of federated learning. In federated learning, some sensors train locally and send the model's variation to the server. Hence, we can use adversarial training to prevent malicious attacks from interfering with this communication between sensors and server. Low complexity neural networks like the ones proposed in this work could also be used in the federated learning approach. Moreover, we believe that others standard building blocks in established communication set up could be replaced by those low complexity neural networks. In this way, the solutions combining ML might result in significant advantages in situations when the assumptions diverge from reality.

Furthermore, we plan to investigate ML-based solutions for massive MIMO sytems. For example, a reinforcement learning scheme can be designed to obtain the set of selected antennas. In this case, the actions might comprise the antennas indices. And instead of choosing just one action as the one with maximum probability, we could select S actions, that is, the ones with highest probabilities. Another possibility is to employ a neural network to estimate the MIMO channel.

Bibliography

[1] LEE, B. M., CHOI, J., BANG, J., KANG, B.-C. "An energy efficient antenna selection for large scale green MIMO systems". In: *International Symposium on Circuits and Systems (ISCAS)*, pp. 950–953, Beijing, China, 2013.

[2] HEATH, R. W., SANDHU, S., PAULRAJ, A. "Antenna selection for spatial multiplexing systems with linear receivers", *IEEE Communications Letters*, v. 5, n. 4, pp. 142–144, Apr 2001.

[3] GAO, X., EDFORS, O., LIU, J., TUFVESSON, F. "Antenna selection in measured massive MIMO channels using convex optimization", *Globecom Workshops (GC Wkshps), 2013 IEEE*, pp. 129–134, Dec 2013.

[4] TIBSHIRANI, R. "Regression shrinkage and selection via the lasso", *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

[5] CHAVES, R. S. "Joint precoding and antenna selection in massive MIMO Systems", *M.Sc. Dissertation*, Mar 2018. COPPE/UFRJ.

[6] LETAIEF, K. B., CHEN, W., SHI, Y., ZHANG, J., ZHANG, Y.-J. A. "The roadmap to 6G: AI empowered wireless networks", *IEEE Communications Magazine*, v. 57, n. 8, pp. 84–90, Aug 2019.

[7] YANG, Z., CHEN, M., WONG, K.-K., POOR, H. V., CUI, S. "Federated learning for 6G: Applications, challenges, and opportunities", *Engineering*, v. 8, pp. 33–41, Jan 2022.

[8] HAMMED, Z. S., AMEEN, S. Y., ZEEBAREE, S. R. "Massive MIMO-OFDM performance enhancement on 5G". In: *2021 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pp. 1–6, Split, Hvar, Croatia, 2021. IEEE.

[9] WONG, K. J., JUWONO, F., REINE, R. "Deep Learning for Channel Estimation and Signal Detection in OFDM-Based Communication Systems", *ELKHA*, v. 14, n. 1, pp. 52–59.

[10] FELLER, W. "Law of large numbers for identically distributed variables", *An introduction to probability theory and its applications*, v. 2, pp. 231–234, 1971.

[11] MALLAT, S. *A wavelet tour of signal processing: the sparse way.* Elsevier/Academic Press, 2009.

[12] DA SILVA, E. A. B., LOVISOLO, L., DUTRA, A. J., DINIZ, P. S. R. "FIR filter design based on successive approximation of vectors", *IEEE Transactions on Signal Processing*, v. 62, n. 15, pp. 3833–3848, May 2014.

[13] JUWONO, F. H., REINE, R. "Future OFDM-based communication systems towards 6G and beyond: machine learning approaches", *Green Intelligent Systems and Applications*, v. 1, n. 1, pp. 19–25, 2021.

[14] PRASAD, R. *OFDM for wireless communications systems.* Boston, Artech House Publishers, 2004.

[15] DINIZ, P. S. R., MARTINS, W. A., LIMA, M. V. S. *Block Transceivers: OFDM and Beyond.* USA, Morgan & Claypool Publishers, 2012.

[16] LIAO, F., WEI, S., ZOU, S. "Deep learning methods in communication systems: A review". In: *Journal of Physics: Conference Series*, v. 1617, p. 012024. IOP Publishing, July 2020.

[17] MENDONÇA, M. O., DINIZ, P. S., FERREIRA, T. N., LOVISOLO, L. "Antenna selection in massive MIMO based on greedy algorithms", *IEEE Transactions on Wireless Communications*, v. 19, n. 3, pp. 1868–1881, Dec 2019.

[18] MENDONÇA, M. O. K., FERREIRA, J. O., DINIZ, P. S. R. "Data Selective Deep Neural Networks for Image Classification". In: *2021 29th European Signal Processing Conference (EUSIPCO)*, pp. 1376–1380, Dublin, Ireland, 2021. IEEE.

[19] FERREIRA, J. O., MENDONÇA, M. O. K., DINIZ, P. S. R. "Data Selection in Neural Networks", *IEEE Open Journal of Signal Processing*, v. 2, pp. 522–534, Aug 2021.

[20] MENDONÇA, M. O. K., MAROTO, J., FROSSARD, P., DINIZ, P. S. R. "Adversarial training with informed data selection". In: *European Signal Processing Conference (EUSIPCO)*, Belgrade, Serbia, 2022.

[21] MENDONÇA, M. O., DINIZ, P. S. "OFDM receiver using deep learning: Redundancy issues". In: *2020 28th European Signal Processing Conference (EUSIPCO)*, pp. 1687–1691, Amsterdam, Netherlands, 2021. IEEE.

[22] DINIZ, P. S., MENDONÇA, M. O. "Zero-Padding OFDM Receiver Using Machine Learning". In: *2021 IEEE Statistical Signal Processing Workshop (SSP)*, pp. 26–30, Rio de Janeiro, Brazil, 2021. IEEE.

[23] MENDONÇA, M. O. K., DINIZ, P. S. R., FERREIRA, T. N. "Machine learning-based channel estimation for insufficient redundancy OFDM receivers using comb-type pilot arrangement", *accepted to Latin-American Conference on Communications (LATINCOM)*, 2022.

[24] MENDONÇA, M. O. K., NETTO, S. L., DINIZ, P. S. R., THEODORIDIS, S. *Machine Learning, Chapter 13, on Signal Processing and Machine Learning Theory edited by Diniz, P. S. R.* Cambridge, UK, Academic Press, 2022.

[25] RUSEK, F., PERSSON, D., LAU, B. K., LARSSON, E. G., MARZETTA, T. L., EDFORS, O., TUFVESSON, F. "Scaling up MIMO: Opportunities and challenges with very large arrays", *IEEE Signal Processing Magazine*, v. 30, n. 1, pp. 40–60, Jan 2013.

[26] WANG, C.-X., HAIDER, F., GAO, X., YOU, X.-H., YANG, Y., YUAN, D., AGGOUNE, H., HAAS, H., FLETCHER, S., HEPSAYDIR, E. "Cellular architecture and key technologies for 5G wireless communication networks", *IEEE Communications Magazine*, v. 52, n. 2, pp. 122–130, Feb 2014.

[27] ANDREWS, J. G., BUZZI, S., CHOI, W., HANLY, S. V., LOZANO, A., SOONG, A. C., ZHANG, J. C. "What will 5G be?" *IEEE Journal on Selected Areas in Communications*, v. 32, n. 6, pp. 1065–1082, Jun 2014.

[28] LARSSON, E. G., EDFORS, O., TUFVESSON, F., MARZETTA, T. L. "Massive MIMO for next generation wireless systems", *IEEE Communications Magazine*, v. 52, n. 2, pp. 186–195, Feb 2014.

[29] MARZETTA, T. L., LARSSON, E. G., YANG, H., NGO, H. Q. *Fundamentals of Massive MIMO*. Cambridge University Press, 2016.

[30] LU, L., LI, G. Y., SWINDLEHURST, A. L., ASHIKHMIN, A., ZHANG, R. "An Overview of Massive MIMO: Benefits and Challenges", *IEEE Journal of Selected Topics in Signal Processing*, v. 8, n. 5, pp. 742–758, Oct 2014.

[31] ALODEH, M., CHATZINOTAS, S., OTTERSTEN, B. "Constructive multiuser interference in symbol level precoding for the MISO downlink channel", *IEEE Transactions on Signal processing*, v. 63, n. 9, pp. 2239–2252, May 2015.

[32] MARZETTA, T. L. "Noncooperative Cellular Wireless with Unlimited Numbers of Base Station Antennas", *IEEE Transactions on Wireless Communications*, v. 9, n. 11, pp. 3590–3600, Nov 2010.

[33] DOMOUCHTSIDIS, S., TSINOS, C. G., CHATZINOTAS, S., OTTERSTEN, B. "Symbol-Level Precoding for Low Complexity Transmitter Architectures in Large-Scale Antenna Array Systems", *IEEE Transactions on Wireless Communications*, v. 18, n. 2, pp. 852–863, Feb 2018.

[34] SANAYEI, S., NOSRATINIA, A. "Antenna selection in MIMO systems", *IEEE Communications Magazine*, v. 42, n. 10, pp. 68–73, Oct 2004.

[35] JOHAM, M., UTSCHICK, W., NOSSEK, J. A. "Linear transmit processing in MIMO communications systems", *IEEE Transactions on Signal Processing*, v. 53, n. 8, pp. 2700–2712, Aug 2005.

[36] ALODEH, M., SPANO, D., KALANTARI, A., TSINOS, C., CHRISTOPOULOS, D., CHATZINOTAS, S., OTTERSTEN, B. "Symbol-level and multicast precoding for multiuser multiantenna downlink: A state-of-the-art, classification and challenges", *IEEE Communications Surveys & Tutorials*, v. 20, n. 3, pp. 1733–1757, May 2018.

[37] GHARAVI-ALKHANSARI, M., GERSHMAN, A. B. "Fast antenna subset selection in MIMO systems", *IEEE Transactions on Signal Processing*, v. 52, n. 2, pp. 339–347, Feb 2004.

[38] HANIF, M., YANG, H. C., BOUDREAU, G., SICH, E., SEYEDMEHDI, H. "Low complexity antenna subset selection for massive MIMO systems with multi-cell cooperation", *2015 IEEE Globecom Workshops*, Dec 2015.

[39] VAZE, R., GANAPATHY, H. "Sub-modularity and antenna selection in MIMO systems", *IEEE Communications Letters*, v. 16, n. 9, pp. 1446–1449, Jul 2012.

[40] BOYD, S., VANDENBERGHE, L. *Convex Optimization.* Cambridge University Press, 2004.

[41] ANTONIOU, A., LU, W. S. *Practical Optimization - Algorithms and Engineering Applications.* Springer, 2007.

[42] GAO, Y., JIANG, W., KAISER, T. "Bidirectional branch and bound based antenna selection in massive MIMO systems". In: *IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 563–568, Hong Kong, China, 2015.

[43] BENMIMOUNE, M., DRIOUCH, E., AJIB, W., MASSICOTTE, D. "Joint transmit antenna selection and user scheduling for massive MIMO systems". In: *Wireless Communications and Networking Conference (WCNC)*, pp. 381–386, New Orleans, LA, USA, 2015.

[44] LIU, A., LAU, V. K. "Joint power and antenna selection optimization for energy-efficient large distributed MIMO networks". In: *IEEE International Conference Communication Systems (ICCS)*, pp. 230–234, Singapore, Singapore, 2012.

[45] BAN, T.-W., JUNG, B. C. "A practical antenna selection technique in multiuser massive MIMO networks", *IEICE Transactions on Communications*, v. 96, n. 11, pp. 2901–2905, Nov 2013.

[46] HAMDI, R., DRIOUCH, E., AJIB, W. "Resource Allocation in Downlink Large-Scale MIMO Systems." *IEEE Access*, v. 4, n. 1, pp. 8303–8314, Nov 2016.

[47] CAETANO, R., DA SILVA, E. A., CIANCIO, A. G. "Video coding using greedy decompositions on generalised bit-planes", *Electronics Letters*, v. 38, n. 11, pp. 1, May 2002.

[48] BJÖRNSON, E., HOYDIS, J., SANGUINETTI, L. "Massive MIMO Networks: Spectral, Energy, and Hardware Efficiency", *Foundations and Trends® in Signal Processing*, v. 11, n. 3-4, pp. 154–655, Nov 2017.

[49] CAIRE, G., SHAMAI, S. "On the achievable throughput of a multiantenna Gaussian broadcast channel", *IEEE Transactions on Information Theory*, v. 49, n. 7, pp. 1691–1706, Jul 2003.

[50] LO, T. K. Y. "Maximum ratio transmission", *IEEE Transactions on Communications*, v. 47, n. 10, pp. 1458–1461, Oct 1999.

[51] PAULRAJ, A., PAPADIAS, C. "Space-time processing for wireless communications", *IEEE Signal Processing Magazine*, v. 14, n. 6, pp. 49–83, Nov 1997.

[52] MALLAT, S., ZHANG, Z. "Matching pursuits with time-frequency dictionaries", *IEEE Transactions on Signal Processing*, v. 41, n. 12, pp. 3397–3415, Dec 1993.

[53] ELAD, M. *Sparse and redundant representations: from theory to applications in signal and image processing*. Springer Science & Business Media, 2010.

[54] LOVISOLO, L., DA SILVA, E. A., DINIZ, P. S. "On the statistics of matching pursuit angles", *Signal Processing*, v. 90, n. 12, pp. 3164–3184, Dec 2010.

[55] MENDONÇA, M. O. K. "Antenna Selection in massive MIMO based on Matching Pursuit", *M.Sc. Dissertation*, Aug 2018. COPPE/UFRJ.

[56] VAN LOAN, C. "How near is a stable matrix to an unstable matrix?" *Cornell University, Technical report*, 1984.

[57] GOLUB, G. H., LOAN, C. F. V. *Matrix Computations*. The John Hopkins University Press, Baltimore and London, 1996.

[58] STEWART, G. W. *Matrix Algorithms Volume 2: Eigensystems*. 2001.

[59] JOSHI, S., BOYD, S. "Sensor selection via convex optimization", *IEEE Transactions on Signal Processing*, v. 57, n. 2, pp. 451–462, Feb 2009.

[60] ASAAD, S., RABIEI, A. M., MÜLLER, R. R. "Massive MIMO with antenna selection: Fundamental limits and applications", *IEEE Transactions on Wireless Communications*, v. 17, n. 12, pp. 8502–8516, Dec 2018.

[61] ASAAD, S., BEREYHI, A., MÜLLER, R. R., RABIEI, A. M. "Asymptotics of transmit antenna selection: Impact of multiple receive antennas". In: *2017 IEEE International Conference on Communications (ICC)*, pp. 1–6, Paris, France, 2017.

[62] DINIZ, P. S. R., MARTINS, W. A., LIMA, M. V. S. "Block Transceivers: OFDM and Beyond", *Synthesis Lectures on Communications*, v. 5, n. 1, pp. 1–206, Jun 2012.

[63] AKAY, E., AYANOGLU, E. "Low complexity decoding of bit-interleaved coded modulation for M-ary QAM". In: *Proc. IEEE ICC*, v. 2, pp. 901–905, Paris, France, 2004.

[64] GRANT, M., BOYD, S., YE, Y. “CVX, Version 1.21 Matlab software for disciplined convex programming”, 2007.

[65] CHEN-NEE CHUAH, TSE, D., KAHN, J., VALENZUELA, R. “Capacity scaling in MIMO wireless systems under correlated fading”, *IEEE Transactions on Information Theory*, v. 48, n. 3, pp. 637–650, Mar 2002.

[66] SHEN, W.-L., LIN, K. C.-J., CHEN, M.-S., TAN, K. “SIEVE: Scalable user grouping for large MU-MIMO systems”. In: *IEEE Conference on Computer Communications (INFOCOM)*, pp. 1975–1983, Kowloon, Hong Kong, 2015.

[67] RUDER, S. “An overview of gradient descent optimization algorithms”, *arXiv preprint arXiv:1609.04747*, 2016.

[68] C. DUCHI, J., HAZAN, E., SINGER, Y. “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization”, *Journal of Machine Learning Research*, v. 12, pp. 2121–2159, Jul 2011.

[69] ZEILER, M. D. “ADADELTA: an adaptive learning rate method”, *arXiv preprint arXiv:1212.5701*, Dec 2012.

[70] KINGMA, D., BA, J. “Adam: A Method for Stochastic Optimization”, *International Conference on Learning Representations*, Dec 2014.

[71] SZEGEDY, C., ZAREMBA, W., SUTSKEVER, I., BRUNA, J., ERHAN, D., GOODFELLOW, I., FERGUS, R. “Intriguing properties of neural networks”, *arXiv preprint arXiv:1312.6199*, 2013.

[72] GOODFELLOW, I. J., SHLENS, J., SZEGEDY, C. “Explaining and harnessing adversarial examples”, *arXiv preprint arXiv:1412.6572*, 2014.

[73] ESTEVA, A., ROBICQUET, A., RAMSUNDAR, B., KULESHOV, V., DEPRISTO, M., CHOU, K., CUI, C., CORRADO, G., THRUN, S., DEAN, J. “A guide to deep learning in healthcare”, *Nature medicine*, v. 25, n. 1, pp. 24–29, 2019.

[74] LIU, Y., RACAH, E., CORREA, J., KHOSROWSHAHI, A., LAVERS, D., KUNKEL, K., WEHNER, M., COLLINS, W., OTHERS. “Application of deep convolutional neural networks for detecting extreme weather in climate datasets”, *arXiv preprint arXiv:1605.01156*, 2016.

[75] DIXON, M., KLABJAN, D., BANG, J. H. “Classification-based financial markets prediction using deep neural networks”, *Algorithmic Finance*, v. 6, n. 3-4, pp. 67–77, Dec 2017.

[76] MADRY, A., MAKELOV, A., SCHMIDT, L., TSIPRAS, D., VLADU, A. "Towards deep learning models resistant to adversarial attacks", *arXiv preprint arXiv:1706.06083*, 2017.

[77] MOOSAVI-DEZFOOLI, S.-M., FAWZI, A., FROSSARD, P. "Deepfool: a simple and accurate method to fool deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

[78] CROCE, F., HEIN, M. "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks". In: *International Conference on Machine Learning*, pp. 2206–2216. PMLR, 2020.

[79] ORTIZ-JIMÉNEZ, G., MODAS, A., MOOSAVI-DEZFOOLI, S.-M., FROSSARD, P. "Optimism in the face of adversity: Understanding and improving deep learning through adversarial robustness", *Proceedings of the IEEE*, v. 109, n. 5, pp. 635–659, Feb 2021.

[80] SU, J., VARGAS, D. V., SAKURAI, K. "One pixel attack for fooling deep neural networks", *IEEE Transactions on Evolutionary Computation*, v. 23, n. 5, pp. 828–841, Jan 2019.

[81] CROCE, F., HEIN, M. "Minimally distorted adversarial examples with a fast adaptive boundary attack". In: *International Conference on Machine Learning*, pp. 2196–2205. PMLR, 2020.

[82] ANDRIUSHCHENKO, M., CROCE, F., FLAMMARION, N., HEIN, M. "Square attack: a query-efficient black-box adversarial attack via random search". In: *European Conference on Computer Vision*, pp. 484–501. Springer, 2020.

[83] DANSKIN, J. M. "The theory of max-min, with applications", *SIAM Journal on Applied Mathematics*, v. 14, n. 4, pp. 641–664, 1966.

[84] DINIZ, P. S. R. "On Data-Selective Adaptive Filtering", *IEEE Trans. Signal Process.*, v. 66, n. 16, pp. 4239–4252, Aug. 2018. ISSN: 1053-587X. doi: 10.1109/TSP.2018.2847657.

[85] MENDONCA, M. O. K., FERREIRA, J. O., TSINOS, C. G., DINIZ, P. S. R., FERREIRA, T. N. "On Fast Converging Data-Selective Adaptive Filtering", *MDPI Algorithms*, v. 12, n. 1, pp. 4, Jan 2019. doi: 10.3390/a12010004.

[86] TSINOS, C. G., DINIZ, P. S. R. "Data-Selective LMS-Newton and LMS-Quasi-Newton Algorithms". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4848–4852, Brighton, UK, May 2019.

[87] DINIZ, P. S. R., MENDONÇA, M. O. K., FERREIRA, J. O., FERREIRA, T. N. "Data-Selective Conjugate Gradient Algorithm". In: *Eusipco: European Signal Processing Conference*, pp. 707–711, Rome, Italy, 2018.

[88] THEODORIDIS, S. *Machine Learning: A Bayesian and Optimization Perspective.* 2nd ed. Orlando, FL, USA, Academic Press, Inc., 2020.

[89] ABU-MOSTAFA, Y. S., MAGDON-ISMAIL, M., LIN, H.-T. *Learning from Data*, v. 4. New York, NY, USA, AMLBook, 2012.

[90] GOODFELLOW, I., BENGIO, Y., COURVILLE, A. *Deep Learning.* London, England, The MIT Press, 2016.

[91] SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SUTSKEVER, I., SALAKHUTDINOV, R. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting", *J. Mach. Learn. Res.*, v. 15, n. 1, pp. 1929–1958, jan. 2014. ISSN: 1532-4435. Disponível em: <`http://dl.acm.org/citation.cfm?id=2627435.2670313`>.

[92] YU, D., SEIDE, F., LI, G., DENG, L. "Exploiting sparseness in deep neural networks for large vocabulary speech recognition". In: *IEEE International Conference on Acoustics, Speech and Signal processing (ICASSP)*, pp. 4409–4412, Kyoto, 2012.

[93] LIU, C., ZHANG, Z., WANG, D. "Pruning deep neural networks by optimal brain damage". In: *Fifteenth Annual Conference of the International Speech Communication Association (ISCA)*, Singapore, 2014.

[94] LUO, J.-H., WU, J., LIN, W. "Thinet: A filter level pruning method for deep neural network compression". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5058–5066, Oct 2017.

[95] VESELỲ, K., HANNEMANN, M., BURGET, L. "Semi-supervised training of deep neural networks". In: *IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 267–272, Olomouc, Czech Republic, 2013.

[96] BLUM, A. L., LANGLEY, P. "Selection of relevant features and examples in machine learning", *Artificial intelligence*, v. 97, n. 1-2, pp. 245–271, Dec 1997.

[97] MACKAY, D. J. “Information-based objective functions for active data selection”, *Neural computation*, v. 4, n. 4, pp. 590–604, Jul 1992.

[98] COHN, D. A., GHAHRAMANI, Z., JORDAN, M. I. “Active learning with statistical models”, *Journal of Artificial Intelligence Research*, v. 4, pp. 129–145, Mar 1996.

[99] PAPOULIS, A., PILLAI, S. U. *Probability, Random Variables, and Stochastic Processes.* Tata McGraw-Hill Education, 2002.

[100] GOLUB, G. H., VAN LOAN, C. F. *Matrix Computations.* The Johns Hopkins University Press, 1996.

[101] FERREIRA, G. J. “Code Repository for Dissertation in GitHub”. Available online at: `https://github.com/Jonathasof/Dissertation`, last accessed: 2022-09.

[102] BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer, 2007.

[103] “Superconductivty Data Data Set”. Available online at: `https://archive.ics.uci.edu/ml/datasets/Superconductivty+Data`, last accessed: 2019-09.

[104] HAMIDIEH, K. “A data-driven statistical model for predicting the critical temperature of a superconductor”, *Computational Materials Science*, v. 154, pp. 346–354, Nov 2018.

[105] “Site Mashable”. Available online at: `https://mashable.com/`, last accessed: 2019-09.

[106] “Online News Popularity Data Set”. Available online at: `https://archive.ics.uci.edu/ml/datasets/Online+News+Popularity`, last accessed: 2022-09.

[107] “Facebook Comment Volume Dataset Data Set”. Available online at: `https://archive.ics.uci.edu/ml/datasets/Facebook+Comment+Volume+Dataset`, last accessed: 2022-09.

[108] KAGGLE. “FIFA 19 Complete Player dataset”. Available online at: `https://www.kaggle.com/karangadiya/fifa19`, last accessed: 2019-09.

[109] YANN LECUN, CORINNA CORTES, C. J. B. “MNIST dataset of handwritten digit”. Available online at: `http://yann.lecun.com/exdb/mnist/`, last accessed: 2019-09.

[110] OF STANDARDS, N. I., (NIST), T. "Emnist Letters dataset". Available online at: `https://www.nist.gov/node/1298471/emnist-dataset`, last accessed: 2019-09.

[111] XIAO, H., RASUL, K., VOLLGRAF, R. "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms", *arXiv preprint arXiv:1708.07747*, 2017.

[112] LOSHCHILOV, I., HUTTER, F. "Online batch selection for faster training of neural networks", *arXiv preprint arXiv:1511.06343*, 2015.

[113] CHANG, H.-S., LEARNED-MILLER, E., MCCALLUM, A. "Active bias: Training more accurate neural networks by emphasizing high variance samples", *arXiv preprint arXiv:1704.07433*, 2017.

[114] KATHAROPOULOS, A., FLEURET, F. "Not all samples are created equal: Deep learning with importance sampling". In: *International Conference on Machine Learning*, pp. 2525–2534. PMLR, 2018.

[115] SHRIVASTAVA, A., GUPTA, A., GIRSHICK, R. "Training region-based object detectors with online hard example mining". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 761–769, Las Vegas, Nevada, 2016.

[116] "Online Video Characteristics and Transcoding Time Dataset Data Set". Available online at: `https://archive.ics.uci.edu/ml/datasets/Online+Video+Characteristics+and+Transcoding+Time+Dataset`, last accessed: 2022-09.

[117] ZHANG, H., YU, Y., JIAO, J., XING, E., EL GHAOUI, L., JORDAN, M. "Theoretically principled trade-off between robustness and accuracy". In: *International Conference on Machine Learning*, pp. 7472–7482. PMLR, 2019.

[118] DING, G. W., LUI, K. Y. C., JIN, X., WANG, L., HUANG, R. "On the Sensitivity of Adversarial Robustness to Input Data Distributions." In: *ICLR (Poster)*, 2019.

[119] DONG, C., LIU, L., SHANG, J. "Data Quality Matters For Adversarial Training: An Empirical Study", *arXiv preprint arXiv:2102.07437*, 2021.

[120] PRECHELT, L. "Early stopping-but when?" In: *Neural Networks: Tricks of the trade*, Springer, pp. 55–69, 1998.

[121] KRIZHEVSKY, A., NAIR, V., HINTON, G. "CIFAR-10 (Canadian Institute for Advanced Research)", Disponível em: <`http://www.cs.toronto.edu/~kriz/cifar.html`>.

[122] HE, K., ZHANG, X., REN, S., SUN, J. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[123] RAGHUNATHAN, A., XIE, S. M., YANG, F., DUCHI, J. C., LIANG, P. "Adversarial training can hurt generalization", *arXiv preprint arXiv:1906.06032*, 2019.

[124] JAVANMARD, A., SOLTANOLKOTABI, M., HASSANI, H. "Precise trade-offs in adversarial training for linear regression". In: *Conference on Learning Theory*, pp. 2034–2078. PMLR, 2020.

[125] ZHANG, J., HE, H., WEN, C. K., JIN, S., LI, G. Y. "Deep learning based on orthogonal approximate message passing for CP-free OFDM", *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 8414–8418, 2019. Brighton, UK.

[126] MARTINS, W. A., DINIZ, P. S. R. "Block-based transceivers with minimum redundancy", *IEEE Transactions on Signal Processing*, v. 58, n. 3, pp. 1321–1333, Mar 2009.

[127] MARTINS, W. A., DINIZ, P. S. R. "LTI transceivers with reduced redundancy", *IEEE Transactions on Signal Processing*, v. 60, n. 2, pp. 766–780, Feb 2011.

[128] RIBEIRO, C. B., CAMPOS, M. L. R., DINIZ, P. S. R. "Time-varying FIR transmultiplexers with minimum redundancy", *IEEE Transactions on Signal Processing*, v. 57, n. 3, pp. 1113–1127, Nov 2009.

[129] YE, H., LI, G. Y., JUANG, B. H. "Power of deep learning for channel estimation and signal detection in OFDM systems", *IEEE Wireless Communications Letters*, v. 7, n. 1, pp. 114–117, Feb 2017.

[130] GAO, X., JIN, S., WEN, C. K., LI, G. Y. "ComNet: Combination of deep learning and expert knowledge in OFDM receivers", *IEEE Communications Letters*, v. 22, n. 12, pp. 2627–2630, Dec 2018.

[131] QIN, Z., YE, H., LI, G. Y., JUANG, B. H. F. "Deep learning in physical layer communications", *IEEE Wireless Communications*, v. 26, n. 2, pp. 93–99, Apr 2019.

[132] DONG, P., ZHANG, H., LI, G. Y., NADERIALIZADEH, N., GASPAR, I. S. “Deep CNN for Wideband Mmwave Massive MIMO Channel Estimation Using Frequency Correlation”, *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 4529–4533, 2019. Brighton.

[133] GOODFELLOW, I., BENGIO, Y., COURVILLE, A. *Deep learning*. London, England, MIT Press, 2016.

[134] O’SHEA, T., HOYDIS, J. “An introduction to deep learning for the physical layer”, *IEEE Transactions on Cognitive Communications and Networking*, v. 3, n. 4, pp. 563–575, 2017.

[135] YE, H., LI, G. Y., JUANG, B.-H. F., SIVANESAN, K. “Channel agnostic end-to-end learning based communication systems with conditional GAN”. In: *IEEE Globecom workshop*, pp. 1–5, 2019 DOI:10.1109/GLOCOMW.2018.8644250.

[136] SAMUEL, N., DISKIN, T., WIESEL, A. “Learning to detect”, *IEEE Transactions on Signal Processing*, v. 67, n. 10, pp. 2554–2564, Feb 2019.

[137] HE, H., WEN, C.-K., SHIN, J., LI, G. Y. “Model-Driven deep learning for joint MIMO channel estimation and signal detection”, *arXiv:1907.09439 [cs.IT]*, pp. 1–27, 2019.

[138] BJÖRNSON, E., GISELSSON, P. “Two applications of deep learning in the physical layer of communication systems”, *arXiv preprint arXiv:2001.03350*, 2020.

[139] COLERI, S., ERGEN, M., PURI, A., BAHAI, A. “Channel estimation techniques based on pilot arrangement in OFDM systems”, *IEEE Transactions on broadcasting*, v. 48, n. 3, pp. 223–229, Sep 2002.

[140] UWAECHIA, A. N., MAHYUDDIN, N. M. “Joint pilot placement and symbol design scheme for sparse channel estimation in OFDM systems”, *Physical Communication*, v. 26, pp. 71–80, Feb 2018.

[141] FANG, M., LI, Y., COHN, T. “Learning how to active learn: A deep reinforcement learning approach”, *arXiv preprint arXiv:1708.02383*, 2017.

[142] CHO, Y. S., KIM, J., YANG, W. Y., KANG, C. G. *MIMO-OFDM wireless communications with MATLAB*. Singapore, John Wiley & Sons, 2010.

[143] HE, H., WEN, C.-K., JIN, S., LI, G. Y. "Model-driven deep learning for MIMO detection", *IEEE Transactions on Signal Processing*, v. 68, pp. 1702–1715, Feb 2020.

[144] MARQUES, A. G. "Guidelines for Evaluation of Radio Transmission Technologies for IMT-2000, Recommendation ITU-R, M. 1225", *Minneapolis: Department of Electrical and Computer Engineering, University of Minnesota*, 1997.

[145] AHMED, A. U., THOMPSON, S. C., ZEIDLER, J. R. "Channel estimation and equalization for CE-OFDM in multipath fading channels", *IEEE Military Communications Conference*, pp. 1–7, 2008. San Diego.

[146] MEINILÄ, J., KYÖSTI, P., JÄMSÄ, T., HENTILÄ, L. "WINNER II channel models", *Radio Technologies and Concepts for IMT-Advanced*, pp. 39–92, 2009. Wiley Online Library.

[147] LIN, Y.-P., PHOONG, S.-M. "Minimum redundancy for ISI free FIR filterbank transceivers", *IEEE Transactions on Signal Processing*, v. 50, n. 4, pp. 842–853, Apr 2002.

[148] LIU, C.-H., PHOONG, S.-M., LIN, Y.-P. "ISI-free block transceivers for unknown frequency selective channels", *IEEE Transactions on Signal Processing*, v. 55, n. 4, pp. 1564–1567, Mar 2007.

[149] GAO, S., DONG, P., PAN, Z., LI, G. Y. "Deep learning based channel estimation for massive MIMO with mixed-resolution ADCs", *IEEE Communications Letters*, v. 23, n. 11, pp. 1989–1993, Aug 2019.

[150] SOLTANI, M., POURAHMADI, V., MIRZAEI, A., SHEIKHZADEH, H. "Deep learning-based channel estimation", *IEEE Communications Letters*, v. 23, n. 4, pp. 652–655, Feb 2019.

[151] GROVER, K., LIM, A., YANG, Q. "Jamming and anti-jamming techniques in wireless networks: A survey", *International Journal of Ad Hoc and Ubiquitous Computing*, v. 17, n. 4, pp. 197–215, 2014.

[152] CLANCY, T. C. "Efficient OFDM denial: Pilot jamming and pilot nulling". In: *2011 IEEE International Conference on Communications (ICC)*, pp. 1–5, Kyoto, Japan, 2011. IEEE.

[153] DURMAZ, M. A., ALAKOCA, H., KURT, G. K., AYYILDIZ, C. "Chirp subcarrier jamming attacks: an OFDM based smart jammer design". In:

2017 16th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net), pp. 1–5, Budva, Montenegro, 2017. IEEE.

[154] BASAR, T. “The Gaussian test channel with an intelligent jammer”, *IEEE Transactions on Information Theory*, v. 29, n. 1, pp. 152–157, 1983.

[155] FLOWERS, B., BUEHRER, R. M., HEADLEY, W. C. “Evaluating adversarial evasion attacks in the context of wireless communications”, *IEEE Transactions on Information Forensics and Security*, v. 15, pp. 1102–1113, Aug 2019.